TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Aleksi Riihiaho

CAD-OHJELMAAN LIITETYN KEHÄRAKENTEIDEN LUJUUSLASKENTAMODUULIN KÄYTTÖLIITTYMÄN SUUNNITTELU JA TOTEUTUS

Diplomityö

Tarkastaja: professori Reijo Kouhia
Tarkastaja ja aihe hyväksytty
Teknisten tieteiden tiedekuntaneuvoston kokouksessa 5. lokakuuta 2016

TIIVISTELMÄ

ALEKSI RIIHIAHO: CAD-ohjelmaan liitetyn kehärakenteiden lujuuslaskentamoduulin käyttöliittymän suunnittelu ja toteutus
Tampereen teknillinen yliopisto
Diplomityö, 57 sivua, 0 liitesivua
Marraskuu 2016
Konetekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Koneiden ja rakenteiden analysointi
Tarkastaja: professori Reijo Kouhia

Avainsanat: käyttöliittymä, olio-ohjelmointi, lujuuslaskenta, elementtimenetelmä

Tämän diplomityön tavoitteena oli suunnitella ja toteuttaa Vertex Systems Oy:n kehittämiin Vertex CAD -suunnitteluohjelmistoihin liitettävän kehärakenteiden lujuuslaskentamoduulin käyttöliittymä. Lujuuslaskentamoduulista tehtiin yksittäinen lisäoptio kolmeen eri Vertex-ohjelmistotuotteeseen, joiden asiakaskunta koostuu mekaniikka-, laitos- ja rakennussuunnittelijoista. Tarve kehärakenteiden lujuuslaskentamoduulille syntyi asiakastarpeesta.

Lujuuslaskentamoduulin käyttöliittymä toteutettiin MVC-ohjelmistoarkkitehtuurityyliä noudattaen. Käyttöliittymä toteutettiin Microsoft Windows -ympäristössä C++-ohjelmointikielellä osana suurempaa Vertex-ohjelmistokokonaisuutta. Käyttöliittymä suunniteltiin ja toteutettiin olio-ohjelmoinnissa yleisesti käytettyjä suunnittelumalleja apuna käyttäen. Työssä tehtiin kirjallisuustutkimusta käyttäjäkokemukseen sekä käyttäjäkeskeiseen suunnitteluun liittyen. Tutkimuksesta saatua tietoa hyödynnettiin käyttöliittymän suunnittelussa ja toteutuksessa.

Kehärakenteiden lujuuslaskentamoduuli rakentuu Vertex CAD -ohjelmistoihin aiemmin kehitetyn elementtimenetelmää hyödyntävän STAFRA-laskentamoottorin päälle. Lujuuslaskentamoduulin käyttöliittymä mahdollistaa rakenteen laskentamallin muodostamisen profiilirakenteen 3d-mallin geometrian perusteella. Solmuverkko voidaan luoda geometrian perusteella automaattisesti ja käyttäjä voi myös muokata sitä interaktiivisesti solmujen ja elementtien tasolla. Laskentamalli koostuu Eulerin-Bernoullin palkkimallin mukaisista elementeistä. Laskentamalli voidaan ratkaista käyttäjän antamilla tuentareunaehdoilla ja kuormituksilla. Ratkaisun tuloksista voidaan lukea laskentamallin rakenneosien rasitussuureita sekä siirtymiä.

ABSTRACT

ALEKSI RIIHIAHO: Design and implementation of user interface for CAD integrated strength analysis module of frame structures
Tampere University of Technology
Master of Science Thesis, 57 pages, 0 Appendix pages
November 2016
Master's Degree Programme in Mechanical Engineering
Major: Analysis of Machines and Structures
Examiner: Professor Reijo Kouhia

Keywords: user interface, object-oriented programming, strength analysis, finite element method

The main objective of this master's thesis was to design and implement a user interface for CAD integrated strength analysis module of frame structures. The strength analysis module will be part of Vertex CAD software which are developed by Vertex Systems Oy. The strength analysis module was developed as an option for three different Vertex software products whose customer base consists of engineers in the fields of machine, plant and building industry. The need for strength analysis module arose from customer needs.

The user interface for strength analysis module was implemented in accordance with MVC software architectural pattern. The user interface was implemented in Microsoft Windows environment with C++ programming language as a part of larger Vertex software package. The user interface was designed and implemented by utilizing design patterns which are commonly used in object-oriented programming. This thesis includes literature research in the fields of user experience and user-centered design. Research results was utilized in the design and implementation of the user interface.

The strength analysis module is based on STAFRA computing engine that uses finite element method. STAFRA is developed earlier by Vertex Systems Oy. The user interface for strength analysis module allows the generation of analysis model based on the geometric 3d model of frame structure. The finite element mesh of analysis model can be generated automatically based on geometric model and user can also interactively modify it in the level of elements and nodes. The analysis model consists of Euler-Bernoulli beam elements. The analysis model can be solved with user defined boundary conditions and loadings. Stresses and deflections of structural members can be read from the results of solution.

ALKUSANAT

Tämä diplomityö on tehty Vertex Systems Oy:n tarjoamasta aiheesta vuoden 2016 tammikuun ja lokakuun välisenä aikana.

Haluan kiittää Vertex Systems Oy:ta mielenkiintoisesta aiheesta sekä koko yrityksen henkilöstöä diplomityön aikana saamastani tuesta ja avusta. Erityisesti diplomityön toteutukseen saamastani avusta haluan kiittää Timo Tulisalmea, Jukka Rissasta ja Marko Mätäsniemeä. Haluan kiittää myös työn ohjaajaa professori Reijo Kouhiaa hyvistä neuvoista sekä ohjauksesta työn aikana.

Lopuksi haluan kiittää perhettäni, ystäviäni ja avopuolisoani Maria saamastani kannustuksesta ja tuesta opintojeni aikana.

Tampereella, 9.10.2016

Aleksi Riihiaho

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

API	engl. Application programming interface, ohjelmointirajapinta
BIM	engl. Building Information Model, rakennuksen tietomalli
C++	olio-ohjelmointikieli
CAD	engl. Computer-Aided Design, tietokoneavusteinen suunnittelu
CFS	engl. Cold-formed steel, kylmämuovattu teräs
FEA	engl. Finite element analysis, elementtimenetelmää hyödyntävä analyysi
FEM	engl. Finite element method, elementtimenetelmä
MVC	engl. model-view-controller, käyttöliittymien suunnittelussa yleinen ohjelmistoarkkitehtuurityyli
STAFRA	engl. Static analysis of frames, Lujuustekniikka Oy:n (nykyisin Vertex Systems Oy) kehittämä avaruuskehien staattiseen analyysiin soveltuva laskentamoottori
XML	engl. Extensible Markup Language, rakenteellinen kuvauskieli tiedon jäsentämiseen sekä tiedonvälitykseen järjestelmien välillä

1. JOHDANTO

Tämän diplomityön toimeksiantajayritys Vertex Systems Oy on vuonna 1977 perustettu suomalainen ohjelmistoyritys. Vertex Systems Oy valmistaa ohjelmistoratkaisuja teollisuuden tarpeisiin monille eri toimijoille, joista tärkeimpinä mainittakoon esimerkiksi metalliteollisuuden kone- ja laitevalmistajat, teolliset talonrakentajat, kalusteiden valmistajat, laitostoimittajat, prosessiteollisuus sekä suunnittelutoimistot. Vertex-ohjelmistoilla on yhteensä noin 18 000 käyttäjää 37 eri maassa. [20]

Tässä diplomityössä suunnitellaan ja toteutetaan Vertex CAD -ohjelmistoihin liitettävän kehärakenteiden lujuuslaskentamoduulin käyttöliittymä. Laskentamoduulin perustana toimii elementtimenetelmää hyödyntävä laskentamoottori. Aiemmin Vertex CAD -ohjelmistoissa kyseistä laskentamoottoria on käytetty automaattiseen ristikkorakenteiden standardin mukaiseen mitoittamiseen. Tässä työssä toteutettavan uuden käyttöliittymän tarkoituksena on laajentaa laskentamoottorin käyttömahdollisuuksia kaikkiin profiilirakenteisiin rajoittumatta mihinkään tiettyihin rakennetyyppeihin.

Tässä diplomityössä toteutettava uusi käyttöliittymä tullaan liittämään kolmeen eri Vertex-ohjelmistotuotteeseen:

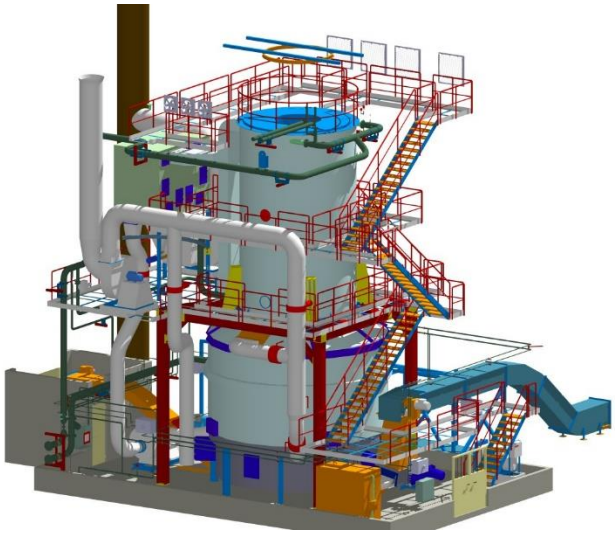
- Vertex G4 -mekaniikkasuunnitteluohjelmistoon
- Vertex G4 Plant -laitos- ja putkistosuunnitteluohjelmistoon
- Vertex BD -rakennussuunnitteluohjelmistoon

Kaikkia näitä kolmea ohjelmistoa yhdistävä tekijä on se, että niillä suunnitellaan erilaisia profiilirakenteita, kuten esimerkiksi konerunkoja, porrastorneja, hoitotasoja, ristikkorakenteita sekä rakennusten runkoja.

Vertex G4 on koneenrakennuksen tarpeisiin tehty 3d-mekaniikkasuunnitteluohjelmisto. Vertex G4 -ohjelmiston profiilirakennetyökaluilla suunnitellaan muun muassa konerunkoja. Kuvassa 1 on torninosturin alavaunu, joka on tyypillinen esimerkki Vertex G4 -ohjelmistolla suunniteltavasta profiilirakenteesta. Vertex G4 -ohjelmiston asiakaskunnasta suurin osa on suomalaisia koneenrakennusyrityksiä.

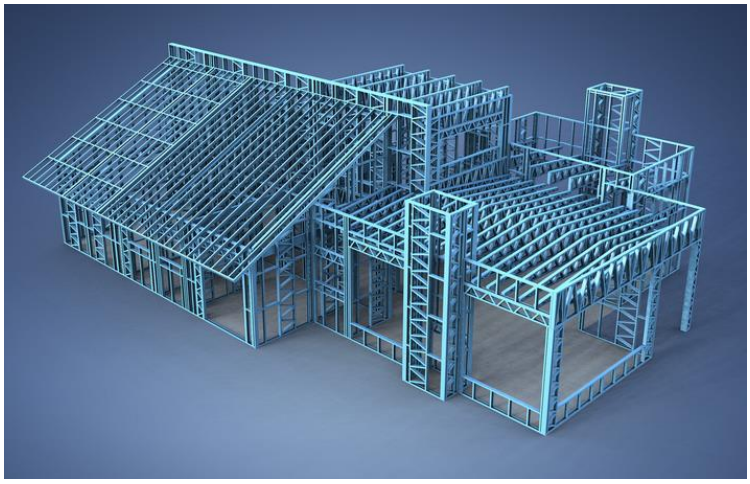
Kuva 1. *Esimerkki tyypillisestä Vertex G4 -ohjelmistolla suunnitellusta profiilirakenteesta. [20]*

Vertex G4 Plant on laitos- ja putkistosuunnitteluun erikoistunut ohjelmisto, jolla onnistuu suurtenkin laitosmallien käsittely. Putkiston kannakoinnit, kanavistot, hoitotasot sekä yleiset profiilirakenteet ovat tyypillisiä Vertex G4 Plant -ohjelmistolla suunniteltavia kohteita. Kuvassa 2 on esimerkki tyypillisestä Vertex G4 Plant -ohjelmistolla suunnitellusta laitoksesta. Vertex G4 Plant -ohjelmiston asiakaskohderyhmään kuuluvat laitos-, instrumentointi- ja prosessisuunnittelua tekevät yritykset.



Kuva 2. *Esimerkki tyypillisestä Vertex G4 Plant -ohjelmistolla suunnitellusta laitoksesta. [20]*

Vertex BD on rakennussuunnitteluohjelmisto arkkitehdeille, rakennesuunnittelijoille sekä teollisille rakentajille. Vertex BD -ohjelmistolla onnistuu älykkään rakennuksen tietomallin (*BIM, Building Information Model*) luominen. Rakennuksen tietomalli muodostaa rakennukseen ja sen rakentamisprosessiin liittyvistä tiedoista digitaalisen kokonaisuuden. Vertex BD -rakennussuunnitteluohjelmiston yksi suurimmista asiakaskohderyhmistä on kylmämuovatusta teräsprofiilista valmistettavien rakennusten suunnittelijat. CFS-profiileista (*Cold-formed steel*) valmistettavia rakennuksia suunnitellaan Vertex BD -ohjelmistolla laajalti ympäri maailmaa, mutta erityisesti Australiassa ja Yhdysvalloissa. Kuvassa 3 on esimerkki tyypillisestä Vertex BD -ohjelmistolla suunnitellusta rakennuksesta, jonka runkorakenne on valmistettu CFS-profiileista. Vertex BD -ohjelmiston Truss Engineering -lisäoptiolla onnistuu automaattinen Australian standardin mukainen mitoittaminen CFS-profiiliristikoille.



Kuva 3. *Esimerkki tyypillisestä Vertex BD -ohjelmistolla suunnitellusta CFS-profiilirakenteesta. [20]*

Näillä kolmella edellä mainitulla ohjelmistolla suunniteltaessa kohdataan usein tilanteita, joissa suunniteltavalle rakenteelle tulisi tehdä lujuustarkasteluja tietyillä kuormituksilla ja tuentareunaehdoilla. Lujuustarkastelujen avulla saadaan selville rakenteen käyttäytyminen kuormituksen alaisena, jonka perusteella voidaan tehdä mitoitusta rakenneosille. Rakenteen käyttäytymistä kuvaavia suureita profiilirakenteille ovat esimerkiksi palkkien tai putkien

- siirtymät,
- leikkausvoimat ja taivutusmomentit poikkileikkauksen pääsuunnissa,
- normaalivoimat,
- akselin suuntaiset normaalijännitykset sekä
- tukireaktiot.

Tässä työssä on tarkoitus suunnitella ja toteuttaa Vertex-ohjelmistoihin asiakastarpeiden pohjalta käyttöliittymä kehärakenteiden lujuuslaskentamoduuliin. Se tulee olemaan suunnittelijalle nopea työkalu esimerkiksi suunnittelunaikaiseen rakennevaihtoehtojen

arvioimiseen. Lujuuslaskentamoduuli toteutetaan yhtenäisenä ohjelmistokomponenttina kaikkiin kolmeen edellä esiteltyyn Vertex-ohjelmistoon. Sen käyttäjinä voivat olla esimerkiksi koneenrunkojen, rakennusten ristikkorakenteiden tai putkistojen suunnittelijat.

Työn alussa käsitellään kehärakenteiden elementtimenetelmän teoriaa, jonka tietämys auttaa ymmärtämään, kuinka palkkielementtejä hyödyntävä lujuuslaskenta suoritetaan. Luvussa kolme esitellään mahdollisimman hyvään käyttäjäkokemukseen ja käytettävyyteen tähtäävän käyttöliittymäsuunnittelun perusperiaatteita. Luvussa neljä perehdytään tämän työn kannalta oleellisiin asioihin liittyen käyttöliittymän ohjelmointiin sekä olio-ohjelmointiin. Luvussa viisi esitellään aiempia Vertex-ohjelmistoissa mukana olleita lujuuslaskentaominaisuuksia sekä määritellään uuden toteutettavan lujuuslaskentamoduulin vaatimuksia. Luvussa kuusi esitellään uuden toteutettavan lujuuslaskentamoduulin käyttöliittymän ominaisuudet sekä sen ohjelmallisen toteutuksen tietorakenne. Luvussa esitellään myös esimerkki lujuuslaskentamoduulin käytöstä. Viimeisessä luvussa tehdään työstä yhteenveto ja esitetään muutamia jatkokehitysideoita.

2. KEHÄRAKENTEIDEN ELEMENTTIMENETELMÄ

Elementtimenetelmä (*Finite element method*, FEM) on matemaattinen menetelmä differentiaali- ja osittaisdifferentiaaliyhtälöiden reuna-arvotehtävien numeeriseen ratkaisemiseen. Elementtimenetelmällä voidaan muuntaa osittaisdifferentiaaliyhtälö algebralliseksi yhtälösysteemiksi, joka on helposti ratkaistavissa. Muunnos tapahtuu diskretoimalla kenttäfunktion äärettömän suuri tuntematon arvojoukko äärelliseksi määräksi solmuarvoja. Diskretoinnin jälkeen solmuarvojen ratkaisu saadaan yhtälösysteemistä. Eri tekniikan aloilla tämänlaisia tehtäviä joudutaan ratkaisemaan paljon, joten elementtimenetelmä onkin nykyään levinnyt muun muassa mekaniikan, termodynamiikan, virtausmekaniikan, murtumismekaniikan, akustiikan ja sähkötekniikan aloille. [9][17]

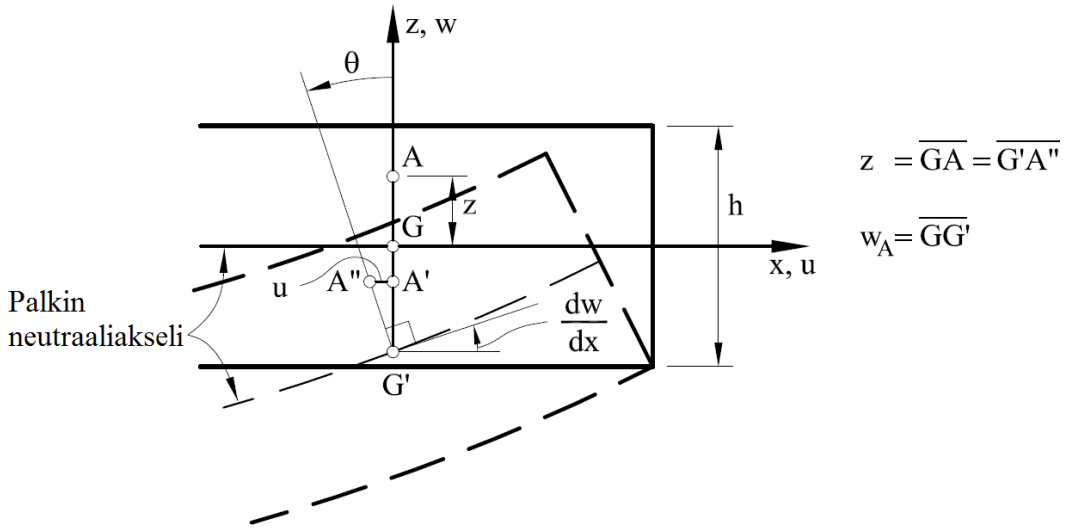
Kehärakenteiden elementtimenetelmässä rajoitutaan ratkaisemaan yksiulotteisista perusrakenneosista koostuvia rakenteita. Tällaisia rakenneosia ovat esimerkiksi sauvat, palkit ja pilarit. Tämän johdosta kehärakenteiden elementtimenetelmä voidaan perustaa suoraan rakenteiden mekaniikan analyyttisten ratkaisumenetelmien teorialle. Tämä tarkoittaa myös sitä, että kehärakenteiden elementtimenetelmä johtaa teknisen taivutusteorian puitteissa tarkkaan ratkaisuun diskretoidun mallin solmukohdissa. [17]

Tämä luku käsittelee kehärakenteiden elementtimenetelmään liittyvää teoriaa. Tämän teorian tietämystä tarvitaan tässä työssä käsiteltävän lujuuslaskentamoduulin käyttämisessä.

2.1 Tekninen taivutusteoria

Teknistä taivutusteoriaa kutsutaan joskus myös klassiseksi palkkiteoriaksi tai sen kehittäjiensä mukaan Eulerin-Bernoullin palkkimalliksi. Se on yksinkertaisin palkin käyttäytymistä kuvaava malli. Teknisen taivutusteorian perusoletukset ovat seuraavat: [9]

1. Palkin poikkileikkaus säilyy tasona taivutuksessa.
2. Palkin poikkileikkaus ei veny taivutuksessa.
3. Palkin poikittainen leikkausmuodonmuutos on merkityksetön.



Kuva 4. *Teknisen taivutusteorian mukainen palkki. Kuva muokattu lähteestä [14].*

Kuvan 4 ja perusoletusten mukaisesti voidaan kirjoittaa siirtymille *u* ja *w* lausekkeet

$$u(x,z) = -z\sin\theta(x) \text{ ja}$$

$$w(x,z) = w_A(x) - (z - z\cos\theta(x)) \,,$$

joissa w_A on palkin neutraaliakselin pisteen *x* Z-suuntainen siirtymäkomponentti ja θ palkin poikkileikkauksen kiertymäkulma. Tekninen taivutusteoria olettaa palkin taipumat pieniksi, jolloin voidaan merkitä $\sin\theta(x) \approx \theta(x)$ ja $\cos\theta(x) \approx 1$. Tällöin siirtymien likimääräisiksi lausekkeiksi saadaan

$$u(x,z) = -z\theta(x) \text{ ja}$$

$$w(x,z) = w_A(x) \,.$$

Kolmannesta oletuksesta seuraa, että palkin poikkileikkauksen kiertymäkulma on sama kuin palkin neutraaliakselin kulmakerroin, jolloin siirtymän *u* lausekkeen voi kirjoittaa seuraavasti:

$$u(x,z) = -z\frac{dw}{dx} \,.$$

2.2 Laskentamalli

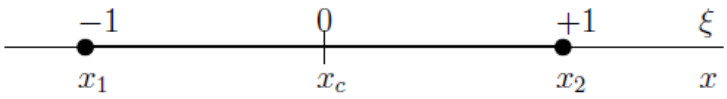
Laskentamallilla tarkoitetaan todellisesta tutkittavasta rakenteesta muodostettua mallia, joka on laskettavissa elementtimenetelmän keinoin. Laskentamalli muodostuu elementeistä, niiden solmuista, tuentareunaehdoista ja kuormituksista. Kehärakenteiden elementtimenetelmässä laskentamalli koostuu yksiulotteisista palkkielementeistä, joilla on tyypillisesti kaksi solmua.

2.2.1 Elementit

Elementtityypin määrittelyyn vaaditaan seuraavat asiat: [9]

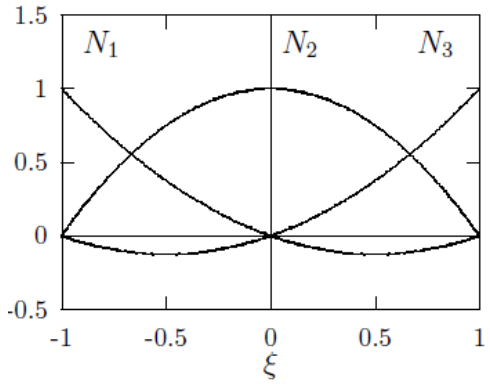
- Elementin geometria
- Muotofunktiot
- Vapausasteiden määrittely

Elementin geometria voi olla lähes mikä tahansa geometrinen kuvio tai kappale, esimerkiksi jana, nelikulmio tai tetraedri. Kehärakenteiden elementtimenetelmässä elementtien geometria on yksiulotteinen, jolloin rajoitutaan janasta tai suljetusta käyrästä muodostuviin elementteihin. Kuvassa 5 on esitetty yksinkertainen janaelementti, jonka paikallinen koordinaatti ξ määritellään välillä $\xi \in (-1,1)$.



Kuva 5. *Yksiulotteinen janaelementti. [9]*

Elementtimenetelmällä muodostetun yhtälöryhmän ratkaisu antaa tuloksia vain laskentamallin solmuissa. Jotta saataisiin tuloksia myös solmujen välillä, eli elementeillä, täytyy elementeille olla määritelty muotofunktiot. Muotofunktiot interpoloivat haluttua suuretta elementin alueella. Tavallisesti muotofunktioina käytetään yksinkertaisia polynomeja. Kuvassa 6 on esitetty kvadraattiset Lagrangen muotofunktiot, jotka ovat esimerkki yleisesti elementtimenetelmässä käytetyistä muotofunktioista. [9]



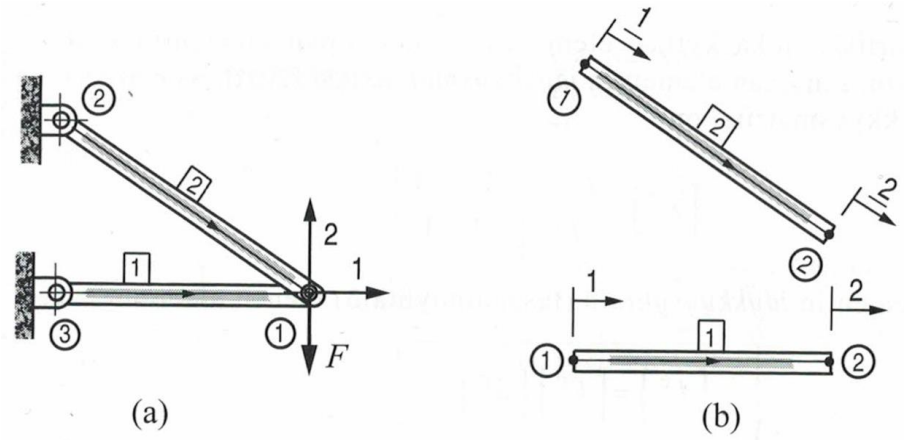
Kuva 6. *Kvadraattiset Lagrangen muotofunktiot. Kuva muokattu lähteestä [9].*

2.2.2 Koordinaatistot

Elementtien väliset solmut voidaan luokitella kahteen kategoriaan: paikallissolmuihin ja rakennesolmuihin. Paikallissolmuilla tarkoitetaan yksittäisen elementin päissä tai nurkissa olevia solmuja, joilla elementit liittyvät toisiin elementteihin. Rakennesolmuilla taas tarkoitetaan koko laskentamallin solmuja. Rakennesolmut yksilöidään yleensä

juoksevalla numeroinnilla, mutta numeroinnin järjestyksellä ei ole laskennan kannalta väliä. Elementin suunnistus määrää sen alku- ja loppupään. Yksittäisen elementin paikallissolmut numeroidaan järjestyksessä pienimmästä suurimpaan alkupäästä lähtien.

Paikalliskoordinaatistolla tarkoitetaan yksittäisen elementin sisäistä koordinaatistoa, jossa X-akseli osoittaa palkin suuntaan alkupäästä loppupäähän päin. Y- ja Z-akselit valitaan yleensä palkin poikkileikkauksen pääsuuntien mukaan. Koko rakenteelle yhteiselle koordinaatistolle käytetään nimitystä rakennekoordinaatisto. Solmumittausjärjestelmällä tarkoitetaan kussakin tilanteessa voimassa olevaa koordinaatistoa, jossa solmumittaus suoritetaan.



Kuva 7. *Laskentamalli (a) ja yksittäiset elementit (b). Kuva muokattu lähteestä [17].*

Kuvassa 7 vasemmalla on esitetty yksinkertaisen sauvarakenteen laskentamalli. Laskentamallin solmumittaus rakennekoordinaatistossa on merkitty nuolilla 1 ja 2. Oikean puoleisessa kuvassa on laskentamallin yksittäiset elementit ja niiden paikalliset solmumittausjärjestelmät. Laskentamallia luodessa yksittäisten elementtien paikalliset solmumittausjärjestelmät tulee muuttaa rakenteen solmumittausjärjestelmän mukaiseksi. Muutos toteutetaan koordinaatistoa kiertämällä.

2.2.3 Solmumittausjärjestelmän kierto

Palkkielementtien solmumittausjärjestelmää joudutaan usein kiertämään, koska palkit voivat osoittaa rakennekoordinaatistossa mielivaltaiseen suuntaan. Jotta sijoittelusummaus voidaan suorittaa, tulee koko rakenteessa olevien yksittäisten palkkielementtien solmumittausten olla samansuuntaiset. Koordinaatiston kierto elementin paikalliskoordinaatistosta rakennekoordinaatistoon ja toisin päin tehdään niin sanotulla koordinaatiston kiertomatriisilla. Kiertomatriisi kootaan siten, että paikalliskoordinaatiston kantavektoreilla $\underline{\mathbf{i}}$, $\underline{\mathbf{j}}$ ja $\underline{\mathbf{k}}$ lausutun vektorin $\underline{\mathbf{v}}$ sekä rakennekoordinaatiston kantavektoreilla **i**, **j** ja **k** lausutun vektorin **v** välillä on lineaarinen yhteys:

$$\begin{bmatrix} \underline{v}_1 \\ \underline{v}_2 \\ \underline{v}_3 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}.$$

2.2.4 Ratkaisukaavat

Yksittäisen elementin tasapainoyhtälöistä on johdettavissa kerroinmatriisi, joka kytkee elementin solmuille kohdistuvat voimat vastaaviksi solmusiirtymiksi. Tätä kerroinmatriisia kutsutaan elementin jäykkyysmatriisiksi. Koko laskentamallin jäykkyyttä kuvaa rakenteen jäykkyysmatriisi. Se kootaan laskentamallin muodostavien elementtien yksittäisistä jäykkyysmatriiseista sijoittelusummauksella. Sijoittelusummauksessa yksittäisten elementtien jäykkyysmatriisien alkiot sijoitetaan rakenteen jäykkyysmatriisin vastaaville paikoille rakenteen solmunumeroinnin mukaisesti. Eri elementeistä tulevat samaan rakennesolmuun liittyvät alkiot summataan yhteen. Sijoittelusummaus edellyttää, että yksittäisten elementtien paikallinen solmumittausjärjestelmä on yhdensuuntainen rakenteen solmumittausjärjestelmän kanssa. Sijoittelusummaus on helposti ohjelmoitavissa ja tehokas tapa koota rakenteen jäykkyysmatriisi tietokonelaskennan kannalta. [17]

Rakenteen laskentamallin kuormitukset kootaan rakennesolmujen kuormitusvektoriin

$$\{\hat{R}\} = \{\hat{P}\} + \{\hat{R}^{ek}\},$$

jossa $\{\hat{P}\}$ on suoraan rakennesolmuille annettu ulkoinen solmukuormitusvektori ja $\{\hat{R}^{ek}\}$ rakennesolmujen ekvivalenttinen solmukuormitusvektori. Rakenteeseen kohdistuva ulkoinen kuormitus jakaantuu usein elementin alueelle, eikä suoraan solmuille, jolloin nämä kenttäkuormitukset täytyy muuntaa solmuvoimiksi. Solmuvoimiksi muunnetut kenttäkuormitukset muodostavat ekvivalenttisen solmukuormitusvektorin. Yksittäisten elementtien ekvivalenttisista solmukuormitusvektoreista saadaan koottua rakennesolmujen ekvivalenttinen solmukuormitusvektori sijoittelusummauksella.

Rakenteen laskentamallin siirtymät saadaan ratkaistua rakennesolmujen tasapainoyhtälöstä

$$\{\hat{R}\} = [K]\{\hat{U}\},$$

jossa $[K]$ on rakenteen jäykkyysmatriisi ja $\{\hat{U}\}$ rakennesolmujen siirtymävektori.

3. KÄYTTÖLIITTYMÄSUUNNITTELUN PERIAATTEITA

Nykyään trendinä on sisällyttää CAD-ohjelmistoihin lisäominaisuutena lujuuslaskentamoduuleja, joiden avulla lujuustarkasteluita voidaan tehdä jo rakenteiden suunnitteluvaiheessa. Kun rakenteen käyttäytyminen kuormitusten alaisena voidaan simuloida tietokoneen avulla, ei rakenteen lujuutta tarvitse varmistaa prototyyppien avulla. Myös tuotekehitysprosessiin kuluva aika ja rahamäärä saadaan pienenemään huomattavasti. Suunnittelunaikaisia lujuustarkasteluja tekevät yleensä henkilöt, jotka eivät välttämättä omaa kovin syvällistä lujuusopin tietämystä, minkä vuoksi lujuuslaskentamoduulien helppokäyttöisyyden merkitys korostuu. [2][16]

Ohjelmistoyrityksen kannattaa panostaa tuotteidensa hyvään ja helppoon käytettävyyteen, sillä se vaikuttaa suoraan tuotteiden laatuun ja vähentää monia kustannuksia. Ottamalla huomioon käytettävyys jo tuotekehitysprojektin alkuvaiheessa saadaan pienennettyä kehityskustannuksia. Kun tiedetään jo projektin alkuvaiheessa, minkälainen käyttöliittymä tyydyttää käyttäjien tarpeet, kehitystyöhön kuluva aika ja kustannukset saadaan minimoitua. Myös tuotteen ylläpitokustannukset saadaan pienemmiksi, kun ohjelmistoa on helppo käyttää, sillä silloin asiakastuen tarve on vähäisempi.

Tässä luvussa perehdytään käyttäjäkokemuksen määritelmään sekä käyttäjäkeskeisen suunnittelun perusperiaatteisiin. Luvun lopussa esitellään muutamia yleisesti hyväksi havaittuja suunnitteluperiaatteita hyvän käytettävyyden edistämiseksi erityisesti ohjelmistojen käyttöliittymien näkökulmasta.

3.1 Käyttäjäkokemus

Käyttäjäkokemuksella tarkoitetaan käyttäjän subjektiivista vaikutelmaa tuotteesta. Käyttäjäkokemus syntyy käyttäjän ollessa vuorovaikutuksessa tuotteen kanssa. Käyttäjäkokemus on siis tunnereaktioita, uskomuksia ja mieltymyksiä, jotka ilmenevät ennen tuotteen käyttöä, käytön aikana ja käytön jälkeen. Täsmällinen määritelmä käyttäjäkokemukselle löytyy ISO 9241-210 -standardista: *”Henkilön havainnot ja vasteet, jotka ovat seurausta tuotteen, järjestelmän tai palvelun käytöstä ja/tai ennakoidusta käytöstä”*. [21]

Käyttäjäkokemukseen voidaan vaikuttaa luonnollisesti tuotteen käytettävyyttä parantamalla. Käytettävyyden kannalta oleellisinta on usein käyttöliittymän hyvä ja huolellinen suunnittelu. Myös käytettävyydelle on luotu standardin mukainen määritelmä (ISO

9241-11) ja se kuuluu näin: *”Tarkoituksenmukaisuus, tehokkuus ja tyytyväisyys, joilla tuotteen määritellyt käyttäjät saavuttavat määritellyt tavoitteet tietyissä käyttöympäristöissä”*. Käytettävyyden lisäksi käyttäjäkokemukseen vaikuttavat myös tuotteen esteettiset ominaisuudet kuten esimerkiksi visuaalisuus ja auditiivisuus. Tuotteeseen liittyvillä oheistarvikkeilla ja -palveluilla on myös vaikutuksensa käyttäjäkokemuksen syntymiseen. Tällaisia voivat olla esimerkiksi tuotteen käyttöohje tai asiakastuki. [21]

Tuotteen käyttöympäristö vaikuttaa merkittävästi käyttäjäkokemuksen syntymiseen. Käyttöympäristö, eli käyttökonteksti, määrittelee millaisessa ympäristössä ja olosuhteissa tuotetta käytetään. Sen voidaan katsoa koostuvan tehtäväkohtaisesta, fyysisestä, sosiaalisesta ja teknisestä ympäristöstä. Teknisestä ympäristöstä esimerkkinä käy vaikkapa laite, jolla kehitettävää ohjelmistoa on suunniteltu käytettävän. Ohjelmiston käyttöliittymää suunniteltaessa on tiedettävä tarkkaan, millaisilla laitteilla loppukäyttäjät tulevat tuotetta käyttämään. Sosiaalisia käyttöympäristöjä ovat esimerkiksi koti ja työpaikka. Kotona tunnelma on usein rentoutunut, kun taas työpaikalla ilmapiiri voi olla virallisempi. [21]

Kuva 8. *Käyttäjäkokemuksen syntyminen käyttäjän ja tuotteen välisessä vuorovaikutuksessa [21]*

Kuvassa 8 on havainnollistettu käyttäjäkokemuksen syntymistä käyttäjän ja järjestelmän vuorovaikutuksen tuloksena. Kuvasta korostuu se, että käyttäjäkokemuksen syntyminen ei ole pelkästään seurausta käytettävyydestä, vaan siihen vaikuttaa oleellisesti myös itse käyttäjä sekä käyttöympäristö.

3.2 Käyttäjäkeskeinen suunnittelu

Käyttäjäkeskeinen suunnittelu on prosessi, jossa tuotteen hyvää käyttäjäkokemusta suunnitellaan hyödyntäen olemassa olevaa käyttäjätietoa. Oikeilta tuotteen loppukäyttäjiltä saatu tieto on usein arvokkaampaa ja oikeellisempaa kuin käyttöliittymää suunnittelevan ohjelmistokehittäjän intuitio. Käyttäjäkeskeisessä suunnittelussa pyritään ymmär-

tämään ihmisen käyttäytymistä ja taitoja sekä hyödyntämään tätä ymmärrystä käyttöliittymän suunnittelussa.

Ohjelmistoprojekteissa yleensäkin tärkeä asia, asiakaslähtöisyys, on erityisen tärkeää käyttäjäkeskeisessä suunnittelussa. Asiakas ja sen tarpeet on tunnettava riittävän hyvin, jotta voidaan suunnitella tuotteen käyttäjäkokemusta oikealle käyttäjäryhmälle. Ohjelmistoprojekteissa asiakkaan tulisikin olla mukana suunnittelussa jo tuotteen vaatimusten määrittelyvaiheessa. Vaatimusten määrittelyssä käyttäjien tarpeiden kartoittaminen koostuu haastatteluista, käyttäjien tarkkailusta sekä kilpailija-analyyseistä. [5]

Käyttäjäkeskeinen suunnittelu on usein luonteeltaan iteratiivista. Asiakkailta ja loppukäyttäjiltä haetaan jo suunnitteluprosessin aikana toistuvasti palautetta siihen asti toteutetuista ratkaisuista. Tarpeeksi ajoissa saatu palaute helpottaa viemään järjestelmää käyttäjäystävällisempään suuntaan. Kun ongelmakohta korjataan jo suunnitteluvaiheessa, prosessin kokonaiskustannukset tulevat hyvin paljon pienemmiksi kuin silloin, jos ongelmakohtiin kiinnitetään huomiota vasta järjestelmän valmistuttua. Iteratiivisen suunnittelun haittapuolena on vaikeus arvioida suunnittelun aikataulua. Suunnittelu- ja toteutuskierroksia saattaa tulla useita ennen kuin järjestelmä saadaan käyttäjää tyydyttävälle tasolle. [5]

Eräs tehokas käyttäjäkeskeisen suunnittelun työkalu on prototypointi. Toteutettavasta tuotteesta tehty konkreettinen hahmotelma paljastaa käytettävyysongelmat paljon paremmin kuin pelkkä paperille kirjoitettu tuotemäärittely. Prototypointi on kustannustehokasta ja nopeaa sekä se auttaa luomaan yhteisen kielen suunnittelijoiden ja käyttäjien välille, kun konkreettista tuotetta pääsee kokeilemaan.

Ohjelmistotuotannon alalla paperiprototyyppi on yleisesti käytetty keino testata järjestelmän käytettävyyttä suunnittelun alkuvaiheessa ennen kuin edes ohjelmointityötä on aloitettu. Paperiprototyyppi voi olla esimerkiksi joukko paperille tulostettuja kuvia tai PowerPoint-esitys, joka mallintaa suunniteltavan järjestelmän käyttöä. Paperiprototypoinnin päämääränä on käyttöliittymän eri näkymien ja niiden välisen vuorovaikutuksen nopea testaaminen. Paperiprototypointi on nopea ja halpa keino testata erilaisia ratkaisuvaihtoehtoja järjestelmän käyttöliittymälle aivan suunnittelun alkuvaiheessa. [5]

Seuraavissa luvuissa 3.3.1 – 3.3.3 käsitellään erilaisia hyvän käyttäjäkeskeisen suunnittelun periaatteita, jotka ovat sovellettavissa käyttöliittymäsuunnitteluun. Suunnitteluperiaatteet soveltuvat myös käytettävyyden heuristiseen arviointiin. [21]

3.2.1 Normanin suunnitteluperiaatteet

Käyttäjäkeskeisen suunnittelun alalla uraa tehnyt professori Donald A. Norman kertoo kirjassaan ”The Design of Everyday Things” (2013) [13] kuinka jokapäiväisten esineiden käytettävyyttä voitaisiin parantaa. Näitä ohjeistuksia on helppo soveltaa myös oh-

jelmistojen käyttöliittymien suunnitteluun. Norman on koonnut ohjeensa viideksi suunnitteluperiaatteeksi:

1. Loogiset kytkennät

Käyttäjälle tulisi selvitä, mikä ohjaimen ja toiminnan välinen kytkentä on. Kytkentä tulisi olla selvä jo ennen kuin käyttäjä on ensimmäistä kertaa käyttänyt ohjainta. Loogisia kytkentöjä suunniteltaessa kannattaa käyttää hyväksi fyysisiä ja tilaan liittyviä analogioita. Esimerkiksi ylöspäin suuntautuva liike ja ylöspäin osoittava nuoli merkitsevät usein lisäystä, kun taas alaspäin suuntautuva liike ja alaspäin osoittava nuoli yhdistetään usein vähentymiseen. Tällaista logiikkaa voidaan käyttää hyväksi esimerkiksi lukumäärää tai mittasuhteita kuvaavissa ohjaimissa.

2. Mahdollisuudet ja rajoitteet

Kun ihminen muodostaa käsitystä tuotteen toiminnasta, hän tekee näkemästään rakenteesta tulkintoja rakenteen mahdollisuuksien ja rajoitteiden perusteella. Mahdollisuudet tarkoittavat eri toimintavaihtoehtoja, joita ovat esimerkiksi käyttöliittymän nappulat, jotka ovat painettavissa. Rajoitteilla puolestaan tarkoitetaan toiminnalle asetettuja rajoituksia, joita ovat esimerkiksi tekstikenttään syötettävän merkkijonon maksimipituus tai harmaannutetut, ei-käytössä olevat käyttöliittymän nappulat.

3. Näkyvyys

Näkyvyydellä tarkoitetaan käyttäjän kannalta oleellisten tietojen ja toimintojen esille tuomista. Käyttöliittymän tulee olla visuaaliselta ilmeeltään sellainen, että käyttäjän on helppo ymmärtää, kuinka tuotetta on tarkoitus käyttää toivotun lopputuloksen saavuttamiseksi. Oikeiden elementtien on oltava käyttöliittymässä näkyviä, jotta käyttäjä saa oikean käsityksen tuotteen toimintatavasta.

4. Palaute

Palaute on käyttäjälle annettavaa tietoa tuotteen toiminnasta. Palautteesta ilmenee, mitä on tehty ja mitä tuloksia saavutettu. Palaute on oleellista onnistuneen toiminnan jälkeen, mutta erityisesti virhetilanteissa palautteen tärkeys korostuu. Olennaista palautteessa on vasteaika. Palautteen on tapahduttava heti käyttäjän toiminnan jälkeen, jotta se osataan yhdistää kyseiseen toimintaan.

5. Virheiden käsittely

Käyttöliittymäsuunnittelun tavoitteena on toki tehdä käyttäjän tekemät virheet mahdottomiksi, mutta se harvoin onnistuu käytännössä. Tämän takia virhetilanteiden käsittely ja niistä palautuminen on keskeistä käyttöliittymäsuunnittelussa. Käyttäjän ei tulisi koskaan syyttää itseään virheistä, sillä se voi johtaa turhautumiseen ja jopa tuotteen hylkäämiseenkin. Käyttäjälle tulee tarjota virhetilanteen sattuessa selkeät ohjeet, miten tilanteesta pääsee pois.

3.2.2 Nielsenin heuristiikat

Käytettävyysasiantuntija Jakob Nielsen esittelee kirjassaan ”Usability Engineering” (1993) [12] kymmenen käytettävyysheuristiikan listan, jonka avulla voi arvioida tuotteen tai käyttöliittymän käytettävyyttä. Lista soveltuu myös uuden tuotteen tai käyttöjärjestelmän suunnittelun periaatteiksi. Nielsenin heuristisen arvioinnin listaan kuuluvat:

1. Järjestelmän tilan näkyvyys
2. Järjestelmän ja todellisen maailman yhteys
3. Käyttäjän kontrolli ja vapaus
4. Johdonmukaisuus
5. Virheiden välttäminen
6. Tunnistettavuus ennen muistamista
7. Käytön joustavuus ja tehokkuus
8. Yksinkertaisuus
9. Virheiden käsittely
10. Opasteet ja dokumentaatio

3.2.3 Shneidermanin kahdeksan kultaista sääntöä

Tietojenkäsittelytieteiden professori Ben Shneidermanin kirjassa ”Designing the user interface: strategies for effective human-computer interaction” (1998) [18] esitellään kahdeksan kultaista sääntöä erityisesti ohjelmistojen käyttöliittymien dialogien suunnitteluun ja arviointiin. Säännöt ovat kuitenkin sovellettavissa ohjenuoriksi myös koko järjestelmän suunnitteluun [21]. Shneidermanin kahdeksan kultaista sääntöä ovat:

1. Noudata yhtenäisyyttä toimintaketjuissa ja toimintatavoissa
2. Tarjoa edistyneille käyttäjille oikoteitä
3. Tarjoa informatiivista palautetta
4. Suunnittele dialogit siten, että ne johtavat lopputulokseen
5. Tarjoa yksinkertaista virheenkäsittelyä
6. Auta käyttäjää toipumaan virhetilanteista
7. Tue käyttäjän kontrollin tunnetta
8. Rajoita käyttäjän lyhytkestoisen muistin kuormitusta

4. KÄYTTÖLIITTYMÄN OHJELMOINTI

Tässä luvussa käsitellään ohjelmistotekniikkaan liittyviä käsitteitä ja malleja, jotka koskevat erityisesti käyttöliittymän ohjelmointia Windows-ympäristössä C++-ohjelmointikielellä. Luvussa painotetaan erityisesti niitä asioita, joiden ymmärtäminen on tässä diplomityössä suunniteltavan ja toteutettavan graafisen käyttöliittymän kannalta oleellisia.

4.1 Olio-ohjelmointi

Ohjelmistotekniikan yhtenä vaikeimpana ongelmana pidetään suurten ohjelmistojen tekemisen monimutkaisuutta. Tietokonelaitteistojen tekniikoiden räjähdysmäinen kehitys on mahdollistanut entistä suurempien ja kompleksisempien ohjelmistojen suorittamisen. Ohjelmistojen tekemiseen tarvittavat työkalut ja menetelmät, kuten ohjelmointikielet, eivät ole kuitenkaan kehittyneet samaa vauhtia laitteistotekniikan kehityksen mukana. Tämän seurauksena on puhjennut niin sanottu ohjelmistokriisi, jolla tarkoitetaan juuri tätä tietokonelaitteistojen ja ohjelmointityökalujen kehityksien epävakaata suhdetta. [15]

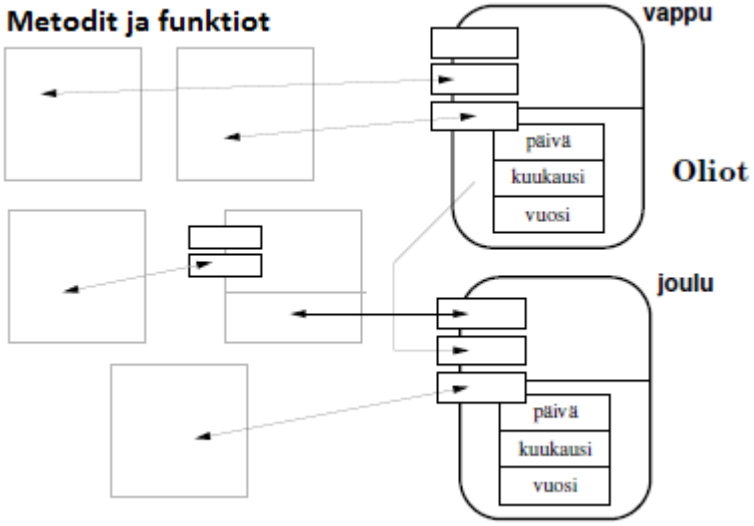
Yhtenä ratkaisuna ohjelmistokriisin ratkaisemiseksi on tarjottu oliokeskeisiä menetelmiä, jotka ovatkin viime aikoina olleet suurimman huomion kohteena ohjelmistoteollisuudessa. Oliokeskeisiä menetelmiä käyttävässä ohjelmoinnissa, eli lyhyemmin olio-ohjelmoinnissa, perusperiaatteina ovat ongelmien jakaminen yhden ihmisen hallittaviin osakokonaisuuksiin sekä yksinkertaistaminen abstrahoimalla. Abstrahoinnin voidaan sanoa tarkoittavan toisiinsa liittyvien asioiden keräämistä yhdeksi järkeväksi toiminnalliseksi kokonaisuudeksi, joka voi olla puhtaasti ajatuksellinen eli abstrakti. [15]

4.1.1 Oliot ja luokat

Olio-ohjelmoinnissa ongelmien jako yhden ihmisen hallittaviin osakokonaisuuksiin tarkoittaa ohjelmakoodin toiminnallisesti sekä tietorakenteellisesti yhteenkuuluvien osien kokoamista olioiksi. Yhden olion rakenne koostuu tietorakenteesta, sitä käsittelevästä ohjelmakoodista sekä olion tietorakenteen käsittelyyn tarkoitetusta julkisesta rajapinnasta. Oliorakenteen toteuttajan vastuulla on olion sisäinen toteutus, eli kuinka sen tietorakenne koostetaan ja kuinka sitä käsitellään. Oliorakenteen toteuttaja suunnittelee oliolle myös julkisen rajapinnan. Julkisen rajapinnan tulisi olla mahdollisimman selkeä ja ymmärrettävä, vaikka sen käyttäjä ei tuntisi olion sisäistä toteutusta lainkaan. Julkinen rajapinta tarkoittaa metodeja, joilla oliota tai sen tilaa voidaan käsitellä olion ulko-

puolelta muualla ohjelmakoodissa. Olio siis kätkee sisäänsä tietorakenteensa ja sen käsittelyn, jolloin olion käyttäjän vastuulle jää ainoastaan olion julkisen rajapinnan oikeaoppinen käyttö. Olion käytön kannalta epäoleellisen tiedon kapseloiminen olion sisälle on yksi tärkeimpiä abstrahoinnin työkaluja ohjelmoinnissa. [15]

C++-kielessä oliot luodaan luokkien avulla. Luokka on määrittely siitä muodostettavan olion tietorakenteelle ja metodeille sekä julkiselle rajapinnalle. Luokka on siis yleinen rakenne, joka edustaa kaikkia kyseisestä luokasta luotuja olioita. Ohjelman ajon aikana ei ole olemassa luokkia, vaan ainoastaan luokan kuvauksen perusteella luotuja olioita. [15]



Kuva 9. *Oliot ja niiden tietorakenne sekä julkinen rajapinta. Kuva muokattu lähteestä [15].*

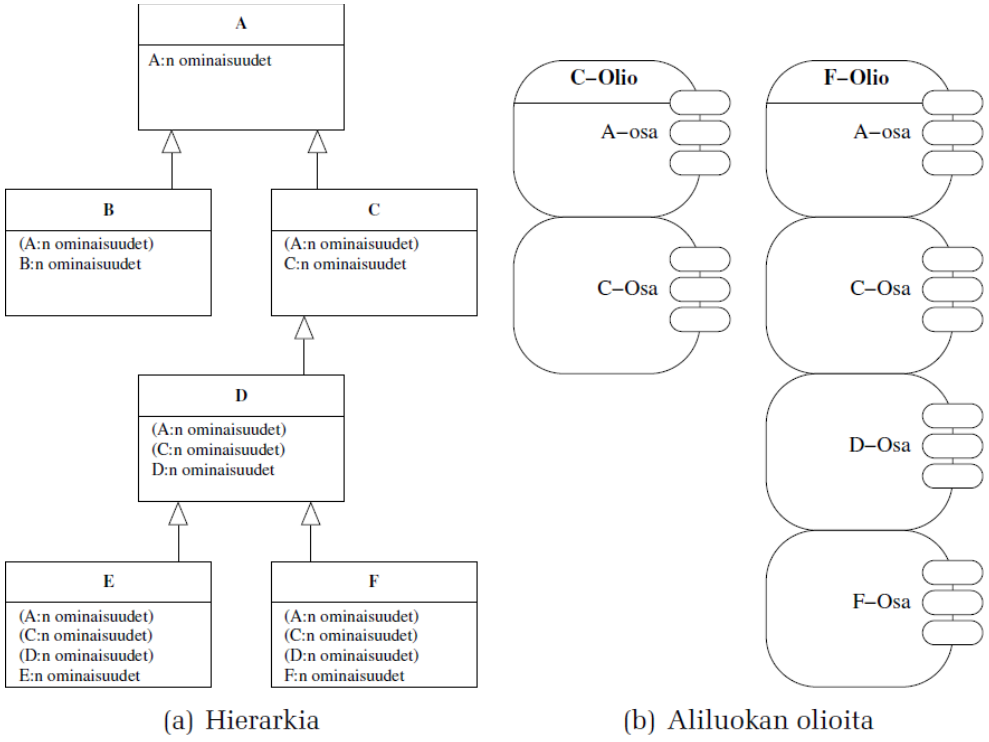
Kuvassa 9 on havainnollistettu tyypillisiä olio-ohjelmoinnissa käytettäviä olioita. Oliot ”vappu” ja ”joulu” ovat samasta luokasta tehtyjä instansseja, ja niiden tarkoitus on esittää päiväystä. Päiväystä esittävän olion tietorakenteeseen kuuluu siis tässä tapauksessa päivä, kuukausi ja vuosi, jotka voidaan ilmaista kokonaislukuina. Olioiden julkisessa rajapinnassa on metodeja, joiden avulla niiden tilaa pystytään muokkaamaan ja kontrolloimaan olioiden ulkopuolelta. Kuvassa olevien päiväystä esittävien olioiden tapauksessa metodit voisivat olla esimerkiksi päiväyksen kasvattaminen halutulla määrällä päiviä tai päiväyksen tulostaminen näytölle tietyllä syntaksilla.

Ohjelmistoja suunniteltaessa olio-ohjelmointi on oiva työkalu ohjelmointiongelman mallintamiseen siten, että se vastaisi mahdollisimman hyvin todellisen maailman ongelmaa. Kun ohjelmassa olevat oliot vastaavat mahdollisimman hyvin todellisen maailman olioita, on monimutkaisenkin tietokoneohjelman rakenne helpommin ymmärrettävissä. Toisaalta ohjelmistoihin suunniteltavien olioiden tulisi olla myös ohjelmoinnin kannalta katsottuna käytännöllisiä ja tehokkaita, jolloin ne eivät välttämättä täsmällisesti vastaa todellisen maailman olioita. Olio-ohjelmointi vaatii ohjelmoijalta hyvää abstraktista ajattelukykyä. [15]

4.1.2 Periytyminen

Yksi tietokoneohjelmien rakenteiden hahmottamisen ja kategorisoimisen avuksi luotu mekanismi on oliokeskeisissä ohjelmointikielissä yleisesti mukana oleva periytyminen. Se tarkoittaa luokkien jaottelua kantaluokkiin ja aliluokkiin niiden yhteisten ominaisuuksien ja sukulaisuussuhteiden perusteella. Periytymisen avulla ohjelmiston olioille on mahdollista luoda hierarkkinen periytymisrakenne, joita esiintyy usein myös todellisessa maailmassa. Periyttämisen avulla voidaan luoda useita samankaltaisia olioita, joilla on hieman toisistaan poikkeava käyttäytyminen, kirjoittamatta kuitenkaan uudestaan olioiden yhtenevän käyttäytymisen toteuttavaa koodia. [15]

Käytännössä C++-ohjelmointikielessä periytymisellä tarkoitetaan sitä, että uusi luokka muodostetaan toisen olemassa olevan luokan pohjalta siten, että uusi luokka sisältää kaikki toisen luokan ominaisuudet. Tällöin alkuperäistä luokkaa, jonka ominaisuudet periytyvät uuteen luokkaan, kutsutaan kantaluokaksi. Uutta, periytettyä luokkaa, kutsutaan aliluokaksi.



Kuva 10. *Periytymishierarkia ja oliot [8].*

Kuvassa 10 on esimerkki luokkien periytymishierarkiasta havainnollistavana kaaviona. Vasemman puolen hierarkiassa nuoli osoittaa aina aliluokasta kantaluokkaan päin. Esimerkin tapauksessa luokka A toimii kantaluokkana kaikille muille luokille. Luokkien ominaisuudet, eli julkisen rajapinnan tarjoamat metodit, periytyvät kuvan vasemman puolen kaavion mukaisesti. Aliluokista luotujen olioiden voidaan ajatella sisältävän kaikki sen periytymishierarkiassa yläpuolella olevien sukulaisluokkien informaation, kuten kuvan oikealla olevassa kuviossa on esitetty.

Aliluokat siis perivät kantaluokkiensa julkisen rajapinnan tarjoamat metodit, ja ne ovat tällöin myös aliluokan itsensä käytettävissä. Joskus on kuitenkin tarpeellista, että aliluokan olio pystyisi käyttämään kantaluokan metodia hieman poikkeavalla tavalla. Aliluokka siis tarvitsisi käyttöönsä kantaluokan julkisen rajapinnan tarjoaman metodin erilaisella toteutuksella. Tällainen tilanne on esimerkiksi silloin, kun aliluokan tietosisältöön kuuluu enemmän informaatiota kuin kantaluokkaan, ja aliluokan käyttämän kantaluokan metodin täytyisi ottaa huomioon myös itse aliluokan rajapinnan kuvaama informaatio. C++-ohjelmointikielessä on mahdollista tehdä kantaluokasta perityllе aliluokan metodille oma toteutus. Se onnistuu, jos metodi on kantaluokassa määritelty virtuaaliseksi. [15]

C++-ohjelmointikielessä on mahdollista määritellä abstrakteja kantaluokkia. Abstraktilla kantaluokalla tarkoitetaan luokkahierarkian sellaista luokkaa, jonka merkityksenä on ainoastaan olla periytymisen kantaluokkana. Niistä ei ole mielekästä tehdä oliota, eikä se ole mahdollistakaan. C++-ohjelmointikielessä luokka on määritelty abstraktiksi kantaluokaksi, mikäli se sisältää vähintään yhden puhtaan virtuaalifunktion. Abstraktissa kantaluokassa puhtaalla virtuaalifunktiolla ei ole toteutusta lainkaan, mutta aliluokissa sen toteutus on pakko määritellä. [15]

4.2 Suunnittelumallit

Olio-ohjelmoinnissa ohjelmiston toiminnallisuus muodostuu suuren oliojoukon yhteistoiminnan vaikutuksesta. Hyvin suunniteltu ohjelmisto sisältää sellaisia oliorakenteita, jotka ovat uudelleenkäytettäviä myös muissa kohteissa. Tällä tarkoitetaan suunnittelun geneerisyyttä. Kirjassa ”Design Patterns: Elements of Reusable Object-Oriented Software” [4] on kerätty hyviksi havaittuja oliosuunnittelun käytäntöjä erilaisiin ohjelmointiongelmiin. Näistä käytännöistä käytetään kirjassa nimitystä suunnittelumalli. Kaikki kirjan käsittelemät suunnittelumallit ovat sellaisia, jotka on havaittu toimiviksi kahden tai useamman toisistaan riippumattoman ohjelmistoprojektihenkilöstön toimesta.

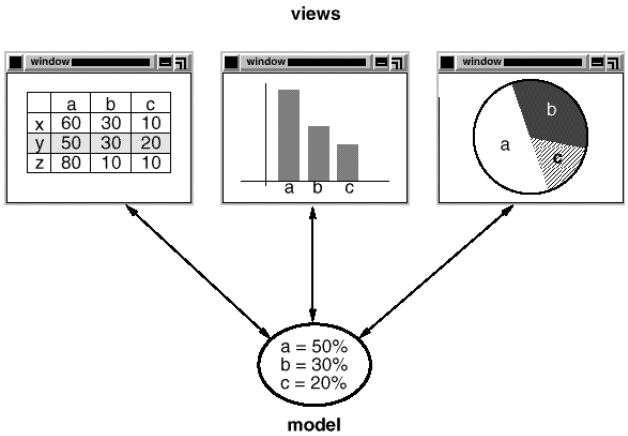
Seuraavissa aliluvuissa esitellään useammasta suunnittelumallista koostuvaa MVC-arkkitehtuuria sekä Visitor pattern -suunnittelumallia. Niiden periaatteita on sovellettu tässä työssä toteutetussa käyttöliittymässä.

4.2.1 MVC-arkkitehtuuri

MVC-arkkitehtuuri (*Model-view-controller)* on kolmesta erilaisesta luokasta koostuva ohjelmistoarkkitehtuurityyli. Se on yleinen suunnitteluperiaate monien käyttöliittymien toteutuksissa. MVC-arkkitehtuurin luokat ovat malli (model), näkymä (view) ja käsittelijä (controller). Sen etuna on käyttöliittymän näkyvän osan toteutuksen erottaminen sen esittämän tietomallin rakenteesta. Tällöin mallin tieto ei ole riippuvainen näkymästä, vaan saman mallin sisältö voidaan esittää usean erilaisen näkymän kautta. Malli ei ole riippuvainen myöskään käsittelijästä, joten se voidaan suunnitella täysin irrallaan käsit-

telijästä. Käsittelijä toimii käyttäjältä tulevien komentojen vastaanottajana muuntaen mallia ja näkymää niiden perusteella. [4]

Kuvassa 11 on havainnollistettu tyypillisen MVC-arkkitehtuurin rakennetta. Se sisältää kolmesta muuttujasta a, b ja c koostuvan mallin sekä kolme erilaista näkymää siihen. Yksinkertaistuksen vuoksi käsittelijä on jätetty kuvasta pois. Taulukko, histogrammi ja ympyräkaavio esittävät mallin tiedon eri tavoin. Malli viestii näkymille, kun siihen tulee muutoksia. Näkymät viestivät mallin kanssa saadakseen pääsyn näytettävään tietoon. Tällä tavoin malli ja näkymät pysyvät aina ajan tasalla. Käsittelijä, jota kuvaan ei ole piirretty, voisi toimia esimerkiksi näppäimistön ja hiiren painallusten tulkitsijana. Koska käsittelijä toteutetaan MVC-arkkitehtuurissa omana luokkanaan, se on helppo vaihtaa toiseen esimerkiksi hieman erilaiseen toteutukseen jotain toista käyttötarkoitusta varten.



Kuva 11. *MVC-arkkitehtuurin malli ja näkymät. [4]*

MVC-arkkitehtuuri koostuu itse asiassa kolmen erilaisen suunnittelumallin hyödyntämisestä. Nämä suunnittelumallit ovat

- Observer-suunnittelumalli, jossa muutos yhden luokan oliossa voi vaikuttaa kuinka moneen tahansa muuhun olioon tarvitsematta mitään tietoa muiden olioiden toteutusyksityiskohdista. Esimerkiksi kuvan 11 tietomallin suhde näkymiin voi olla tällainen.

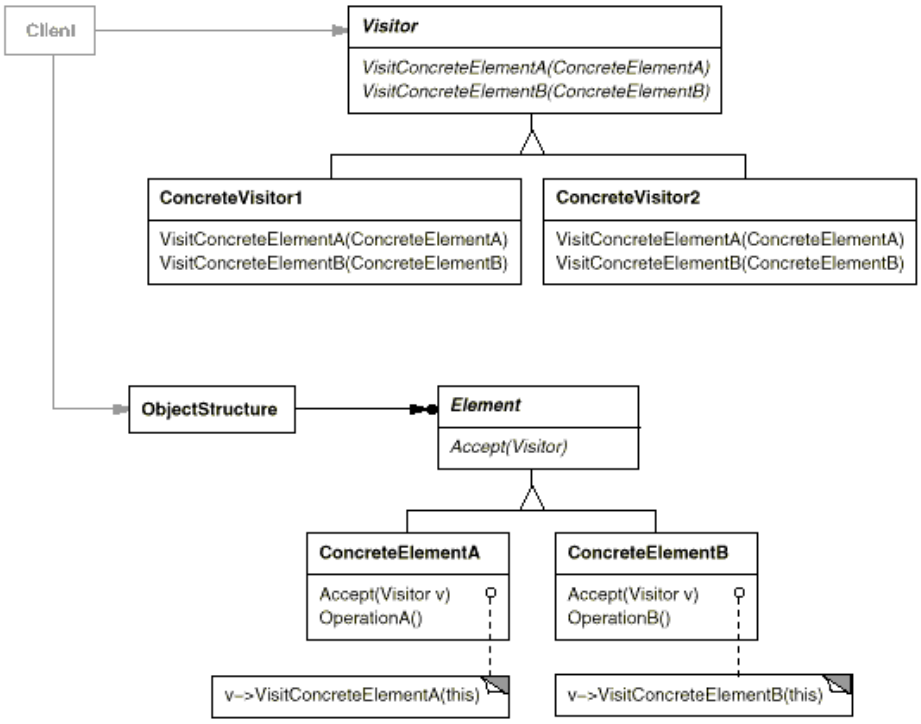
- Composite-suunnittelumalli, jossa yksittäisiä olioita käsitellään samalla tavoin kuin koosteolioita. Koosteolioilla on yhteinen kantaluokka yksittäisten olioiden kanssa, ja ne sisältävät ryhmän yksittäisiä olioita. Esimerkiksi kuvan 11 näkymät voisivat olla koosteolioita, jolloin niiden sisällä voisi olla vielä muita näkymiä.

- Strategy-suunnittelumalli, jossa jokin algoritmi toteutetaan luokkana. MVC-arkkitehtuurin tapauksessa algoritmina toimii käsittelijä-luokka, joka huolehtii käyttäjän antamien komentojen toteuttamisesta. Käsittelijä-luokkia voi olla useita ja niitä on mahdollista vaihtaa jopa ohjelman ajon aikana. Tällä tavoin saadaan muutettua näkymän vastetta käyttäjän syötteisiin helposti ainoastaan käsittelijä-oliota vaihtamalla.

4.2.2 Visitor pattern -suunnittelumalli

Visitor pattern -suunnittelumalli mahdollistaa tietorakenteen ja siihen liittyvien metodien irrottamisen toisistaan. Suurin käytännön hyöty tästä on se, että uusien tietorakennetta läpikäyvien metodien toteuttaminen onnistuu helposti muuttamatta itse tietorakennetta. Visitor pattern -malli on hyödyllisin silloin, kun tietorakenteen tiedetään pysyvän kohtuullisen muuttumattomana, mutta tietorakenteeseen liittyviä uusia metodeja halutaan mahdollisesti toteuttaa myöhemmin lisää. Se on erityisen käyttökelpoinen myös silloin, kun tietorakenne koostuu useista erilaiset rajapinnat sisältävistä luokista, joille kaikille halutaan tehdä samantyyppisiä, mutta hieman toisistaan poikkeavia metodeja. Metodeja ei tällöin tarvitse toteuttaa tietorakenteen sisälle, vaan ne voidaan ulkoistaa omiksi luokikseen. [4]

Visitor pattern -suunnittelumallin perusidea on se, että tietorakenne itse huolehtii alkioidensa läpikäymisen, ja erillinen `Visitor`-luokka suorittaa halutun operaation yhdelle tietorakenteen alkiolle. Kuvassa 12 on esitetty luokkakaavio, joka havainnollistaa suunnitteluperiaatteen toimintaa.

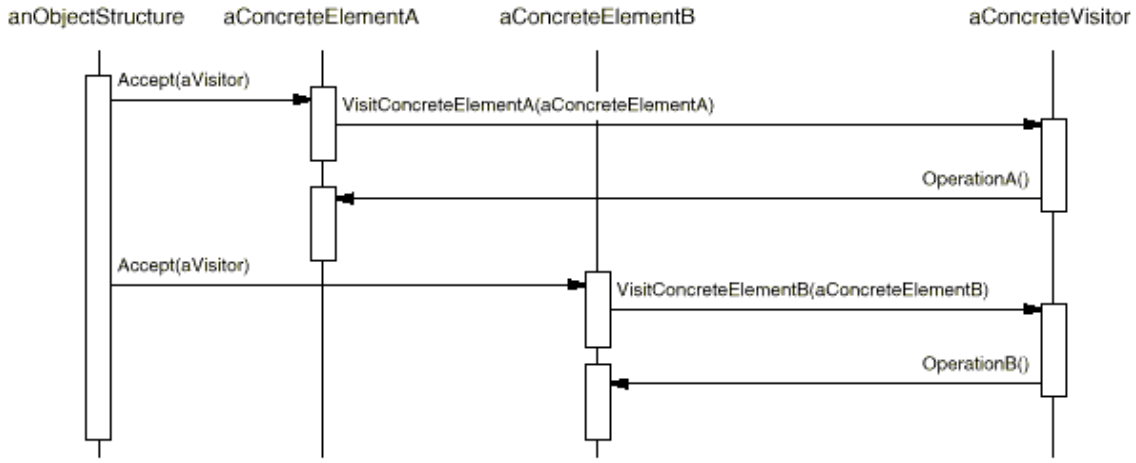


Kuva 12. *Visitor pattern -suunnittelumalli. [4]*

Visitor pattern -suunnittelumallin voidaan katsoa koostuvan seuraavista viidestä komponenttityypistä:

- **`Visitor`-luokka,** joka esittelee vastaavan `visit`-metodin jokaiselle yksittäiselle tietorakenteen alkion luokalle. `Visitor`-luokka toimii kantaluokkana `ConcreteVisitor`-luokille.

- **`ConcreteVisitor`-luokka,** joka toteuttaa suoritettavan operaation yksittäiselle tietorakenteen alkiolle. Jokaista tietorakenteen alkion luokkaa kohden on myös oma metodinsa, joiden toteutus voi olla erilainen. `ConcreteVisitor`-luokan olio voi myös säilöä tietoa itseensä käydessään läpi tietorakennetta.
- **`Element`-luokka,** joka esittelee `accept`-metodin. Parametrinaan `accept`-metodi saa `Visitor`-luokan olion. `Element`-luokka toimii kantaluokkana kaikille tietorakenteen alkioiden luokille.
- **`ConcreteElement`-luokka,** joka vastaa yksittäisen tietorakenteen alkion luokkaa. Tämä luokka toteuttaa `accept`-metodin, joka ikään kuin päästää `Visitor`-luokan olion tekemään operaationsa `ConcreteElement`-luokan olioon.
- **Tietorakenne,** joka koostuu `Element`-luokasta periytetyistä olioista. Siihen kohdistuvat operaatiot ovat `Visitor`-luokasta periytettyjen `ConcreteVisitor`-olioiden toteuttamia. Tietorakenteen muoto on täysin vapaa. Se voi olla esimerkiksi lista, puu tai graafi.



Kuva 13. *Visitor pattern -suunnittelumallin mukainen tietorakenteen läpikäynti. [4]*

Kuva 13 havainnollistaa Visitor pattern -suunnittelumallin mukaista tietorakenteen läpikäyntiä. Kuvan tietorakenne on yksinkertaistettu siten, että se koostuu vain kahdesta alkiosta `ConcreteElementA` ja `ConcreteElementB`. Tietorakenteen alkiot kutsuvat `Accept`-metodissa parametrina saadun `Visitor`-luokan olion `visit`-metodia, joka taas saa parametrikseen käsiteltävänä olevan tietorakenteen alkion. Tälle alkiolle `Visitor`-olio suorittaa itse operaation.

5. VERTEXIN LUJUUSLASKENTAOMINAISUUKSIEN UUDISTAMINEN

Vertexillä ryhdyttiin vuoden 2015 lopussa projektiin, jonka tavoitteena oli kehittää Vertex-ohjelmistojen lujuuslaskentaominaisuuksia kilpailukykyisempään suuntaan seuraaviin vuonna 2016 marraskuussa julkaistaviin pääversioihin.

Tässä luvussa esitellään aikaisempia Vertex-ohjelmistoissa mukana olleita lujuuslaskentaominaisuuksia sekä määritellään asiakasvaatimukset, joiden pohjalta Vertex-ohjelmistojen lujuuslaskentaominaisuuksia tullaan kehittämään.

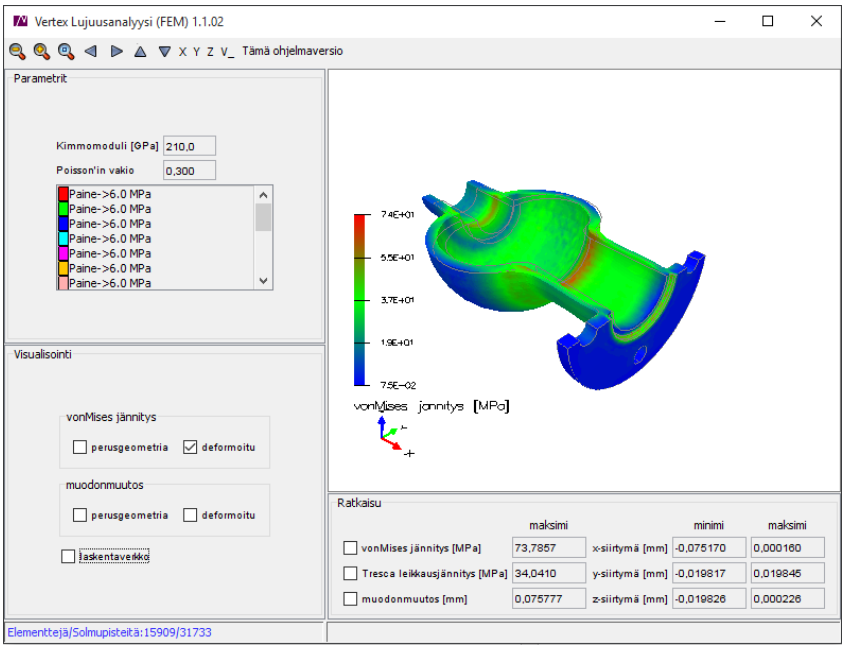
5.1 Asiakasvaatimukset

Aiemmin Vertex-ohjelmistoissa on ollut seuraavia lujuuslaskentaominaisuuksia:

- Yksittäisille kappaleille soveltuva tilavuuselementtejä hyödyntävä lujuusanalyysi Vertex G4 -ohjelmistossa.
- Ristikkorakenteiden tai yksittäisen palkin mitoittaminen kantavien rakenteiden suunnittelustandardien mukaisesti Vertex BD -ohjelmistossa Truss Engineering -lisäoptiolla. Tuettuna on Australiassa käytössä oleva kylmämuovattujen teräsrakenteiden suunnittelustandardi AS/NZS 4600:2005 Cold-formed steel structures.

Seuraavissa luvuissa 5.1.1 ja 5.1.2 selvitetään, minkälaisia tarpeita Vertex G4, G4 Plant ja BD -asiakkailla olisi ohjelmistojen lujuuslaskentaominaisuuksien näkökulmasta sekä esitellään ohjelmistojen nykyisiä lujuuslaskentaominaisuuksia.

5.1.1 Vertex G4 ja G4 Plant



Kuva 14. *Vertex G4 -ohjelmiston yksittäisen tilavuusmallin lujuusanalyysi.*

Kuvassa 14 näkyy Vertex G4 -ohjelmistossa mukana oleva yksittäisen tilavuusmallin lujuuslaskentamoduuli. Se on kehitetty yhteistyössä laskennallisen teknologian palveluja tarjoavan Numerola Oy:n kanssa ja se pohjautuu heidän kehittämään ohjelmistokomponenttiin. Lujuuslaskentamoduulin heikkoutena on se, että sillä voidaan analysoida ainoastaan yksittäisiä kappaleita. Kokoonpanojen analysoiminen on mahdollista ainoastaan tekemällä kokoonpanomallista yksittäinen kappale boolean-operaatiolla. Tällöin kokoonpanojen kappaleiden väliset kontaktipinnat häviävät ja mahdollisuus analysoida kosketuksissa olevia kappaleita katoaa. Laskentamallin elementtiverkon luominen on myös tuottanut ongelmia ohuilla rakenteilla. Esimerkiksi joillekin putkipalkeille elementtiverkon luominen ei onnistu lainkaan.

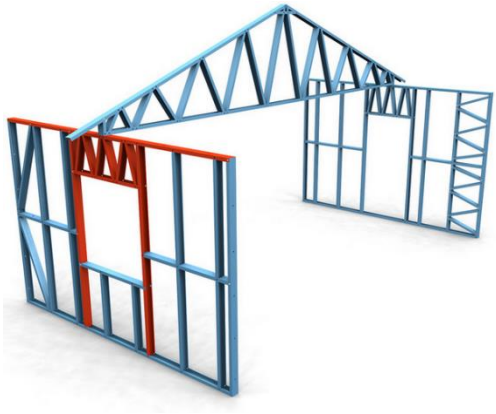
Vertexin asiakaskunnalta kyseltiin, minkälaisia kokemuksia heillä oli nykyisestä Vertex G4 -ohjelmistossa olevasta lujuuslaskentamoduulista. Asiakkailta saatiin paljon kehitysehdotuksia ja toiveita tuleviin versioihin. Alla on listattu eniten esiin nousseita toiveita:

- Kehärakenteiden siirtymien, rasitusten ja tukivoimien laskenta
- Nurjahdusanalyysit
- Putkistojen jännitysanalyysit
- 3d- ja laskentamallin välinen keskinäinen synkronointi malleja päivitettäessä
- Laskentamallin tekeminen ilman 3d-mallia
- Ohutlevyosien analysoiminen kuorielementeillä
- Kokoonpanomallien analysoiminen tilavuuselementeillä
- Väsymisanalyysi

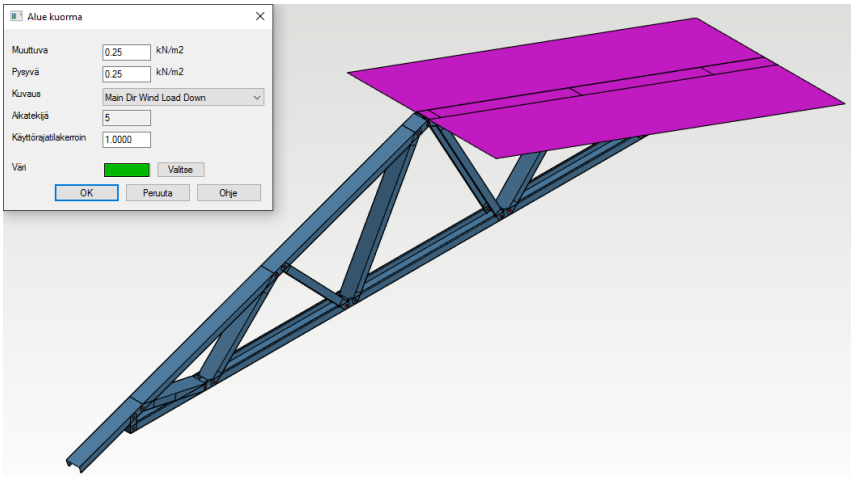
Vertex G4 Plant -laitossuunnitteluohjelmiston asiakaskunnasta on tullut myös toiveita mahdollisuuteen mitoittaa suunnittelustandardien mukaisesti esimerkiksi laitosten hoitotasoja.

5.1.2 Vertex BD

Tällä hetkellä Vertex BD tukee ainoastaan ristikkorakenteiden sekä yksittäisen palkin suunnittelustandardin mukaista mitoitusta. Asiakaskunnasta on tullut toiveita, että Vertex BD -ohjelmistolla voisi mitoittaa myös muunlaisia rakenteita. Esimerkiksi kuvassa 15 näkyvä ikkuna-aukkopalkki on tyypillinen kohde, jota asiakkaat haluaisivat mitoittaa.



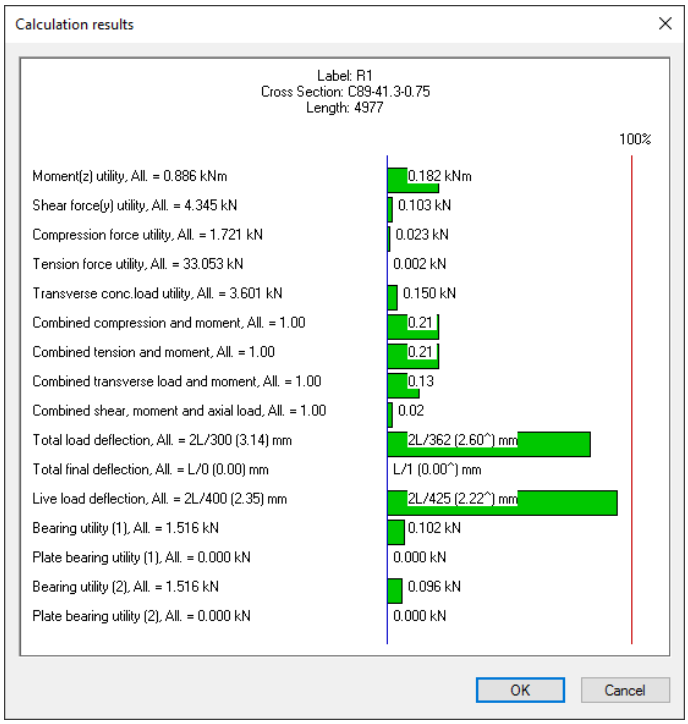
Kuva 15. *CFS-profiilirakenteinen ikkunan aukkopalkki. [20]*



Kuva 16. *Aluekuorma Vertex BD -ohjelmiston Truss Engineering -lisäoptiossa.*

Vertex BD -ohjelmiston Truss Engineering -lisäoptiolla onnistuu CFS-profiilista valmistettujen ristikkorakenteiden mitoitus AS/NZS 4600:2005 Cold-formed steel structures -standardin mukaisesti. Mitoitusmoduulin käyttö on tehty rakennesuunnittelijan kannalta mahdollisimman yksinkertaiseksi. Käyttäjän tarvitsee vain osoittaa malliin tulevat kuormat ja niiden tyypit. Rakenteesta luotavaa laskentamallia ei näytetä käyttäjälle missään vaiheessa, vaan se luodaan automaattisesti. Käyttäjä voi asettaa alue- tai

pistekuormat mihin kohtaan mallia tahansa kuvan 16 osoittamalla tavalla. Kuormat muutetaan ratkaisun aikana laskentamallin vastaaviksi solmukuormituksiksi. Laskentamallin tuennat muodostetaan automaattisesti rakenteen geometrian perusteella. Kun mitoitus on suoritettu, käyttäjä voi tarkastella rakenneosien kapasiteetteja standardin määrittelemille rasitussuureille kuvan 17 mukaisesti.



Kuva 17. *Mitoituksen raportti rakenneosan käyttöasteista Vertex BD -ohjelmiston Truss Engineering -lisäoptiossa.*

Tällainen yhdelle rakennetyypille räätälöity mitoitustyökalu on suunnittelijalle todella tehokas ja helppokäyttöinen. Mitoitus onnistuu tällöin vaivatta myös lujuuslaskentaan tai elementtimenetelmään perehtymättömältäkin käyttäjältä. Toisaalta, jos rakenteen laskentamallissa on jotain korjattavaa ja laskenta ei onnistu, käyttäjällä ei ole mitään mahdollisuuksia korjata sitä tai edes saada selville syytä, miksi FEM-laskenta ei onnistu. Mitoitustyökalun vaarana on se, että mitoitettava rakenne saattaa jäädä alimitoitetuksi, jos laskentamalli ei vastaa todellisen rakenteen käyttäytymistä riittävän tarkasti. Käyttäjä ei siis koskaan voi olla täysin varma laskentamallin oikeellisuudesta.

5.2 Uudet lujuuslaskentamoduulit

Asiakasvaatimusten pohjalta tehtiin päätös, että seuraaviin vuonna 2016 marraskuussa julkaistaviin Vertex-ohjelmistojen pääversioihin tullaan lisäämään kolme eri tavalla hinnoiteltavaa lisäoptiota:

- Tilavuusmallien FEM-analyysi Vertex G4 ja G4 Plant -ohjelmistoihin. Tämä laskentamoduuli tulee korvaamaan entisen Numerola Oy:n kanssa yhteistyössä tehdyn lujuuslaskentamoduulin.
- Kehärakenteiden FEM-analyysi Vertex G4, G4 Plant ja BD -ohjelmistoihin.
- Kehärakenteiden mitoitus Vertex BD -ohjelmistoon, joka vaatii myös kehärakenteiden FEM-analyysi -lisäoption.

Seuraavissa luvuissa 5.2.1 – 5.2.4 esitellään STAFRA-laskentamoottoria sekä kaikkia kolmea uutta kehitettävää lujuuslaskentamoduulia. STAFRA-laskentamoottori toimii kaikkien uusien lujuuslaskentamoduulien perustana.

5.2.1 STAFRA-laskentamoottori

STAFRA *(Static analysis of frames)* on Lujuustekniikka Oy:n (nykyisin Vertex Systems Oy) kehittämä avaruuskehien statiikan ongelmien ratkaisuun tarkoitettu laskentamoottori. Se perustuu luvussa 2 esiteltyyn kehärakenteiden elementtimenetelmään ja sen elementit ovat Eulerin-Bernoullin palkkimallin mukaisia elementtejä. Eulerin-Bernoullin palkkimallista poiketen STAFRA ottaa huomioon myös leikkausjännityksistä aiheutuvat liukumat tehollisten leikkauspinta-alojen avulla. Tehollliset leikkauspinta-alat palkin paikalliskoordinaatiston Y- ja Z-suunnissa saadaan kaavoilla

$$A_y = A/\zeta_y \text{ ja } A_z = A/\zeta_z \tag{10}$$

palkkielementin paikalliskoordinaatiston X-akselin yhtyessä poikkileikkausten pintakeskiöihin sekä Y- ja Z-akselit poikkileikkausten pääneliöakseleihin. Kertoimet ζ_y ja ζ_z aiheutuvat leikkausjännityksen epätasaisesta jakaantumisesta poikkileikkauksen alueelle ja ne ovat riippuvaisia poikkileikkauksen muodosta. STAFRA rajoittuu elementtien materiaalien osalta homogeenisiin, lineaarisesti kimmoisiin ja isotrooppisiin materiaaleihin. [6]

STAFRA-laskentamoottoriin sisältyy toiminnallisuus profiilin poikkileikkausarvojen laskemiseen 3d-mallin geometriasta. Laskettavia poikkileikkausarvoja ovat

- pinta-ala,
- jäyhyysmomentit molemmissa pääsuunnissa ja
- taivutusvastukset molemmissa pääsuunnissa.

STAFRA kykenee myös tunnistamaan yleisimmät poikkileikkausmuodot 3d-mallin geometriasta. Mikäli poikkileikkausmuoto on tunnistettavissa, voidaan lisäksi laskea poikkileikkauksen

- vääntöjäyhyys,
- vääntövastus ja

- teholliset leikkauspinta-alat pääsuunnissa.

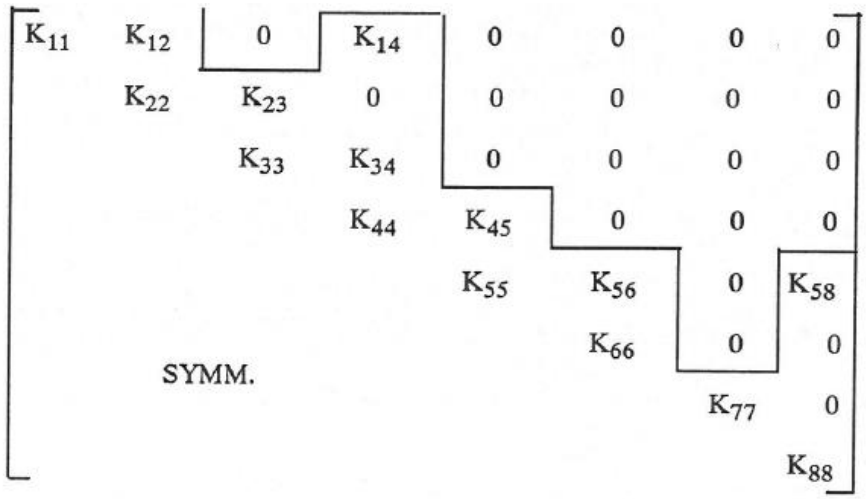
STAFRA-laskentamoottorin tunnistamia poikkileikkausmuotoja ovat

- I-profiili
- L-profiili
- U-profiili
- C-profiili
- Z-profiili
- T-profiili
- Ympyrä
- Ellipsi
- Pyöreä putki
- Suorakaide
- Ontto suorakaide

Mikäli poikkileikkauksen muoto ei ole tunnistettavissa, vääntöjäyhyyden arvoksi oletetaan 1. ja 2. pääsuunnan jäyhyysmomenttien summa. Tällöin myös leikkauspinta-alojen arvoja ei lasketa, mistä johtuen leikkausmuodonmuutoksia ei huomioida laskennassa.

Materiaaliominaisuudet STAFRA-laskentamoottori saa 3d-mallin nimiketiedoista. 3d-mallin nimiketietoihin on Vertex-ohjelmistoissa tallennettu viittaus materiaalitietokannan kenttään, josta FEM-laskennassa tarvittavat materiaaliominaisuudet saadaan.

STAFRA käyttää jäykkyysmatriisin tallentamiseen tietokoneen muistissa niin sanottua profiilimuotoa. Elementtimenetelmän käyttämät jäykkyysmatriisit ovat tavallisesti symmetrisiä ja nauhamaisia. Nauhamaisuudella tarkoitetaan sitä, että iso osa matriisin alkioista on nollia, ja muun kuin nollan sisältävät alkiot sijaitsevat lähellä matriisin diagonaalia nauhamaisena muodostelmana. Matriisin nauhamaisuuden vuoksi sitä ei kannata tallentaa tietokoneen muistiin tavanomaisena 2-ulotteisena taulukkona, vaan riittää kun tallennetaan alkioita nauhan leveyden verran. Lisäksi, kun jäykkyysmatriisit ovat symmetrisiä, riittää ainoastaan puolikkaan nauhan tallentaminen. Tämä matriisien tallennustapa käyttää tietokoneen muistia huomattavasti tavanomaista tapaa taloudellisemmin, ja elementtimenetelmän yhtälöryhmien ratkaiseminen on myös selvästi nopeampaa. [1]



Kuva 18. *Jäykkyysmatriisin tallentaminen tietokoneen muistiin profiilimuodossa. [6]*

Kuvassa 18 on havainnollistettu matriisin tallentamista profiilimuodossa. Ainoastaan symmetrisen matriisin puolikkaan sisälle piirretyn profiiliviivan alapuolella olevat alkiot tallennetaan. Profiiliviiva piirretään matriisin sisälle siten, että sen yläpuolelle jää ainoastaan nolla-alkioita. Tallennus tapahtuu kahden taulukon avulla:

- **Taulukko A,** johon tallennetaan alkiot järjestyksessä sarakkeittain. Tässä tapauksessa sarakkeella tarkoitetaan diagonaalialkiosta ylöspäin olevia alkioita aina viimeiseen alkioon ennen profiiliviivaa.
- **Taulukko MAXA,** johon tallennetaan matriisin diagonaalialkioiden paikat taulukossa A.

Vapausasteiden globaali numerointi, ja sitä kautta myös solmunumerointi, vaikuttavat jäykkyysmatriisin nauhanleveyteen. Jotta jäykkyysmatriisi saadaan tallennettua tietokoneen muistiin mahdollisimman pieneen tilaan, tulee solmunumeroinnin olla optimaalinen. STAFRA käyttää vapausastenumeroinnin optimointiin Gibbs-Poole-Stockmeyer -algoritmia. [6]

STAFRA on käytössä Vertex BD -ohjelmiston nykyisessä ristikkorakenteiden mitoituslaskennassa Truss Engineering -lisäoptiossa. Tulevaisuudessa sitä tullaan hyödyntämään kaikissa uusissa lujuuslaskentamoduuleissa.

5.2.2 Tilavuusmallien FEM-analyysi

Tilavuusmallien FEM-analyysi tulee korvaamaan vanhan Numerola Oy:n kanssa yhteistyössä tehdyn lujuuslaskentamoduulin. Parannuksena aiempaan, uudessa Tilavuusmallien FEM-analyysissä tutkittavana kappaleena voi olla myös kokoonpanomalli, joka sisältää useita osia. Kokoonpanomallin osien välille tulee myös mahdollisuus antaa kontaktireunaehtoja. Uuden lujuuslaskentamoduulin käyttöliittymä tullaan uudistamaan visuaalisemmaksi ja intuitiivisemmaksi.

Uusi Tilavuusmallien FEM-analyysi tulee käyttämään STAFRA-laskentamoottoria jäykkyysmatriisien kolmioinnin ja siirtymien laskennan osalta. Elementtiverkko tullaan muodostamaan automaattisesti 3d-mallin geometrian pohjalta. 4- tai 10-solmuisista tetraedrielementeistä koostuvan elementtiverkon luomiseen käytetään apuna Spatial Corporationin 3D Mesh toolkit -ohjelmakirjastoa. [19]

Käyttäjälle Tilavuusmallien FEM-analyysi tulee olemaan tehokas työkalu suunnittelun aikana tehtävään pienimuotoiseen lujuusanalyysiin. Esimerkiksi nostokorvakkeiden suunnittelun aikainen lujuustarkastelu tulee onnistumaan nopeasti ja helposti uudella Tilavuusmallien FEM-analyysilla.

5.2.3 Kehärakenteiden FEM-analyysi

Kehärakenteiden FEM-analyysi tulee olemaan STAFRA-laskentamoottoria käyttävä palkkielementeistä koostuvien laskentamallien statiikan ongelmia ratkaiseva lujuuslaskentamoduuli. Laskentamallin luominen tullaan tekemään Vertex-ohjelmistolla mallinnettujen profiilirakenteiden perusteella helposti ja käyttäjäystävällisesti.

Käyttäjälle Kehärakenteiden FEM-analyysi tulee olemaan tehokas työkalu suunnittelun aikana tehtävään pienimuotoiseen lujuusanalyysiin profiilirakenteille. Esimerkiksi kattoristikon paarteen maksimitaivutusmomentin tai laitoksen putkilinjan maksimisiirtymän ratkaiseminen tulee onnistumaan nopeasti ja helposti uudella Kehärakenteiden FEM-analyysilla.

5.2.4 Kehärakenteiden mitoitus

Kehärakenteiden mitoitus tulee olemaan lisäoptio Kehärakenteiden FEM-analyysiin. Se tekee analysoitavalle rakenteelle rasitusten laskennan lisäksi myös kantavien rakenteiden suunnittelustandardien mukaisen mitoituksen. Se hyödyntää aiemmin pelkästään Vertex BD -ohjelmistossa Truss Engineering -lisäoptiossa käytössä ollutta DesignEngine-ohjelmistokomponenttia. Vuonna 2016 julkaistavaan pääversioon tuettuna tulee olemaan Australiassa käytössä oleva suunnittelustandardi AS/NZS 4600:2005 Cold-formed steel structures.

Käyttäjälle Kehärakenteiden mitoitus tulee olemaan tehokas työkalu nopeaan profiilirakenteen mitoittamiseen. Esimerkiksi rakennuksen seinän runkorakenteessa olevan ikkunan aukkopalkin mitoittaminen standardin mukaisesti tulee onnistumaan nopeasti ja helposti uudella Kehärakenteiden mitoitus -optiolla.

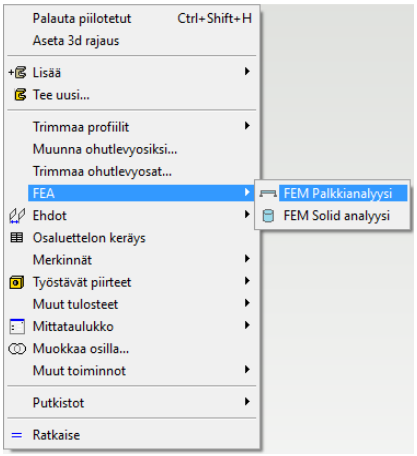
6. KEHÄRAKENTEIDEN LUJUUSLASKENTA-MODUULIN KÄYTTÖLIITTYMÄ

Tässä luvussa perehdytään kehärakenteiden lujuuslaskentamoduulin käyttöliittymän suunnitteluun ja toteutukseen. Se julkaistaan vuonna 2016 Vertex-ohjelmistojen päävers-ioissa. Lujuuslaskentamoduuli tulee olemaan osana Vertex G4, G4 Plant sekä BD -ohjelmistoja. Käyttöliittymä toteutetaan C++-ohjelmointikielellä osana suurempaa Ver-texin ohjelmistokokonaisuutta.

Luvussa 6.1 perehdytään käyttöliittymän ominaisuuksiin yleisellä tasolla ja luvussa 6.2 esitellään käyttöliittymän toteuttavan ohjelmakoodin rakennetta. Luvussa 6.3 tarkastel-laan esimerkkiä kehärakenteiden lujuuslaskentamoduulin käytöstä.

6.1 Ominaisuudet

Kehärakenteiden lujuuslaskentamoduuli käynnistyy Vertex-ohjelmistojen 3d-mallinnustilan kontekstivalikon kautta kuvan 19 mukaisesti. Lujuuslaskentamoduulin käynnistyttyä käyttäjä siirtyy FEA-tilaan *(Finite element analysis)*, jossa lujuuslasken-taan liittyviä toimintoja on mahdollista suorittaa. FEA-tilassa käyttäjän on mahdollista luoda tutkimuksia, jotka kuvaavat erilaisia rakenteen kuormitustilanteita. Yksi tutkimus sisältää tiedon siihen kuuluvista profiileista, elementtiverkon, tuentareunaehdot sekä kuormitukset. Käyttäjä voi ratkaista tutkimuksia ja tarkastella rakenteen rasituksia anne-tuilla kuormitus- ja tuentareunaehdoilla. Tulosten perusteella on helppo vertailla esi-merkiksi erilaisten rakennesuunnitelmien käyttäytymistä kuormituksen alaisena.



Kuva 19. *FEA-tilaan siirtyminen 3d-mallinnustilan kontekstivalikon kautta.*

FEA-tilassa mallin geometrian muokkaus on estetty käyttäjältä. Tämä johtuu siitä, että on järkevää erottaa lujuuslaskennan tekeminen geometrian mallintamisesta, koska ne ovat selkeästi suunnitteluprosessin kaksi erilaista työvaihetta. FEA-tilasta on toki helppo tulla pois ja tehdä mallin geometriaan muutoksia. Geometrian muuttaminen aiheuttaa haasteita tutkimusten tiedon ylläpidossa. Jos esimerkiksi mallin yksittäistä palkkia pidennetään, tuleeko palkin keskellä olevan pistekuorman liikkua suhteessa pituudenmuutokseen vai säilyttää absoluuttisen paikkansa? Yleispätevin ratkaisu on antaa käyttäjälle ilmoitus, että geometria on muuttunut, eikä tutkimuksiin liittyvä tieto enää ole ajan tasalla. Näin meneteltiin myös tässä lujuuslaskentamoduulissa.

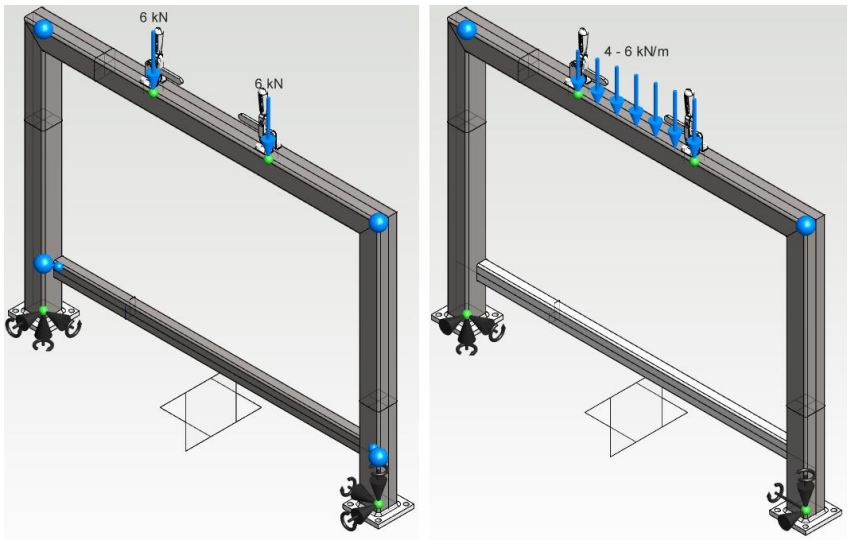
Kehärakenteiden lujuuslaskentamoduulin käyttöliittymä koostuu kolmesta osa-alueesta:

- 3d-mallissa esitettävästä grafiikasta,
- tutkimuspuusta ja
- toiminnoista.

3d-mallissa esitettävä grafiikka ja tutkimuspuu toimivat näkyminä käsiteltävään laskentamalliin. FEA-tilassa suoritetut toiminnot kohdistetaan aina aktiivisena olevan tutkimuksen laskentamalliin.

6.1.1 3d-mallin grafiikka

3d-mallissa esitettävän grafiikan tarkoituksena on havainnollistaa yksinkertaisesti ja mahdollisimman visuaalisesti yhden käsiteltävänä olevan tutkimuksen sisältö. Käyttäjälle pyritään näyttämään kaikki oleellinen tieto graafisesti.



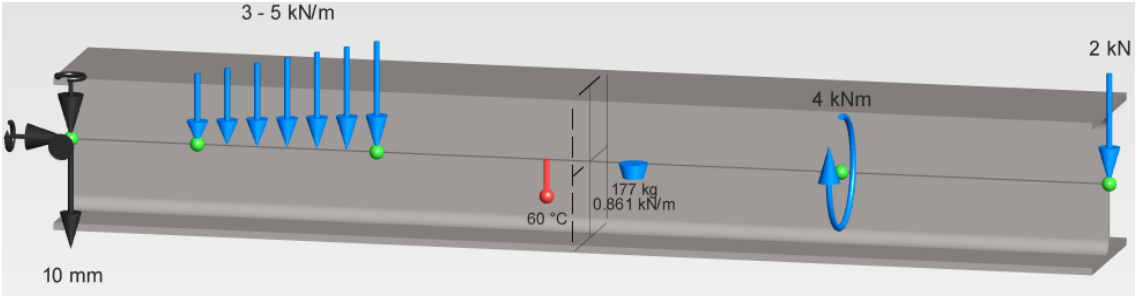
Kuva 20. *Kaksi erilaista tutkimusta samasta 3d-geometriasta.*

Siirryttäessä FEA-tilaan 3d-mallin aktiiviseen tutkimukseen kuulumattomat osat väritetään valkoisiksi. Aktiivista tutkimusta vaihtamalla valkaistut osat vaihtuvat. Tällöin nähdään yhdellä silmäyksellä, mistä osista laskentamalli on luotu. Käyttäjällä ei ole

myös mahdollisuutta kohdistaa FEA-tilan toimintoja osiin, jotka eivät kuulu aktiiviseen tutkimukseen. Kuvassa 20 on esitetty kaksi erilaista tutkimusta samasta 3d-mallista.

Laskentamalliin liittyvät grafiikkaobjektit piirretään suoraan 3d-mallin geometrian päälle. Tällöin synnytetään visuaalisesti looginen yhteys todellisen rakenteen ja siitä luotavan laskentamallin välille.

Käyttäjällä on myös mahdollisuus tarttua lähes kaikkiin 3d-mallissa oleviin laskentamalliin liittyviin grafiikkaobjekteihin hiirellä klikkaamalla. Täten laskentamallin sisältö, kuten palkkielementtien solmut ja niihin liittyvät kuormitukset ovat käyttäjän valittavissa suoraan 3d-mallista.

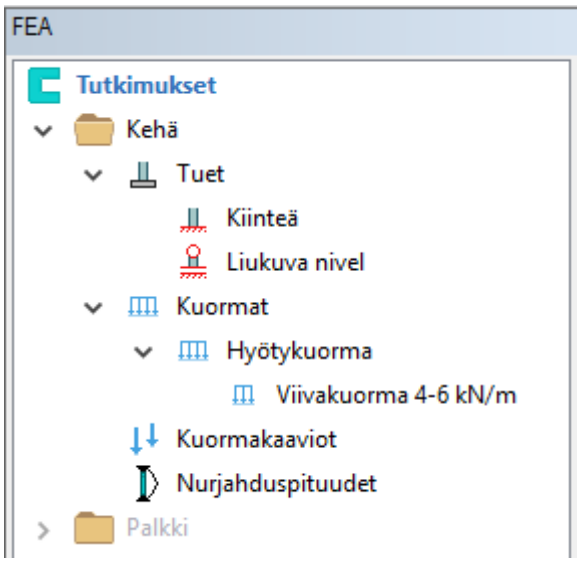


Kuva 21. *Palkin 3d-malli ja siihen liittyvän laskentamallin grafiikkaobjekteja.*

Laskentamallin grafiikkaobjektien ulkoasulla on tavoitteena viestiä käyttäjälle visuaalisesti mahdollisimman selkeästi, mitä kukin objekti esittää. Esimerkiksi kuormituksia kuvaavat nuoliobjektit kuvaavat mihin kohtaan rakennetta kuormitus kohdistuu ja mihin suuntaan se vaikuttaa. Symbolien väreillä pyritään yhdistämään samaan kategoriaan kuuluvat asiat ja toisaalta erottelemaan eri asioita tarkoittavat symbolit. Esimerkiksi kaikki voimaa tai momenttia kuvaavat kuormitussymbolit ovat vaaleansinisiä, kun taas muodoltaan kuormitusta muistuttavat tuentasymbolit ovat mustia. Kuvassa 21 näkyy kaikkien erityyppisten kuormitusten 3d-grafiikkasymbolit. Vasemmalta päin lueteltuna ne ovat pakkosiirtymä, jatkuva voima, lämpökuorma, omapaino, pistemomentti ja pistevoima.

6.1.2 Tutkimuspuu

Tutkimuspuu on 3d-mallin grafiikan lisäksi toinen näkymä käsiteltävään laskentamalliin. Se sijaitsee omassa ikkunassaan 3d-mallin ikkunasta irrallaan. Tutkimuspuun tarkoituksena on esittää tiivistetyssä muodossa kaikki yhteen 3d-malliin liittyvät tutkimukset.



Kuva 22. *Tutkimuspuu.*

Yksittäiseen tutkimukseen sisältyvästä tiedosta tutkimuspuussa esitetään ainoastaan tärkeimmät, eli tuennat, kuormat, kuormakaaviot sekä nurjahduspituudet. Tutkimuspuun tarkoituksena onkin toimia kokonaisten tutkimusten hallintapaneelina. Kaikki kokonaisten tutkimusten käsittelyyn liittyvät toiminnot käynnistyvät tutkimuspuun kautta. Kuvassa 22 näkyy tutkimuspuun hierarkkinen rakenne.

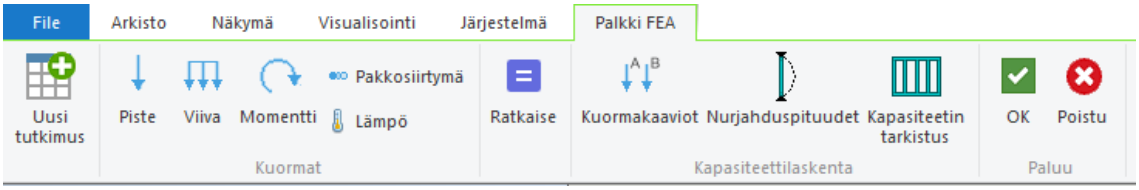
6.1.3 Toiminnot

Tutkimusten sisällön käsittelyyn on toteutettu laaja kirjo erilaisia toimintoja. Toiminnot käynnistyvät joko valintanauhan kautta tai 3d-mallin ikkunan päällä hiiren oikeaa nappia painettaessa avautuvan tilannekohtaisen valikon kautta.

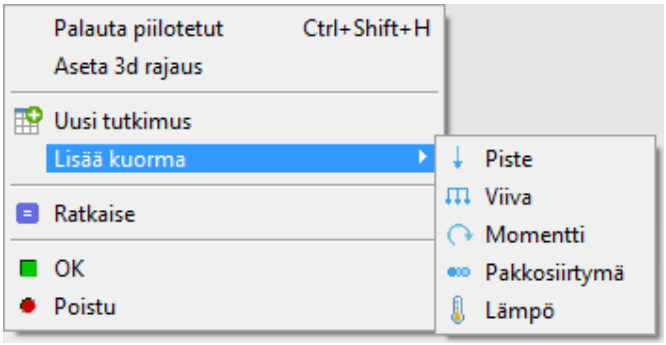
Kehärakenteiden lujuuslaskentamoduulin käyttöliittymässä on kolme erilaista valintanauhaa, joista kaksi ovat tilannekohtaisia. Valintanauhat ovat:

- **FEA-tilan valintanauha**, joka on näkyvillä aina kun käyttäjä on FEA-tilassa.
- **Palkkien tilannekohtainen valintanauha,** joka on näkyvillä yhden tai useamman 3d-mallin profiilin ollessa valittuna.
- **Solmujen tilannekohtainen valintanauha,** joka on näkyvillä yhden tai useamman aktiivisen tutkimuksen solmun ollessa valittuna.

Tilannekohtaisesti näkyviin tulevat valintanauhat auttavat käyttäjää huomaamaan heti, mitä toimintoja valittuina oleville objekteille on mahdollista tehdä.



Kuva 23. *FEA-tilan valintanauha.*

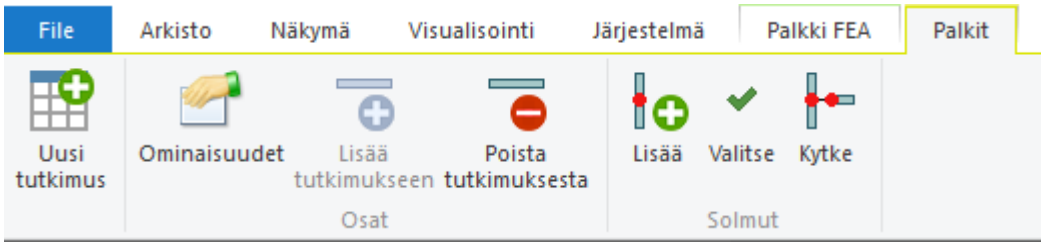


Kuva 24. *FEA-tilan tilannekohtainen valikko 3d-mallin ikkunassa.*

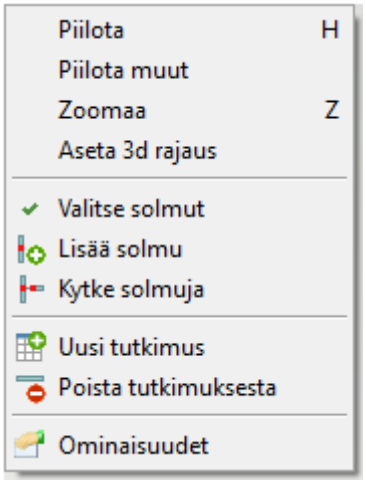
Kuvissa 23 ja 24 näkyvät FEA-tilassa käytettävissä olevat ei-tilannekohtaiset toiminnot. Niiden kuvaukset ovat koottuna taulukkoon 1. Valintanauhan kapasiteettilaskenta-ryhmässä olevat toiminnot ovat saatavilla ainoastaan Kehärakenteiden mitoitus -lisäoptiossa.

Taulukko 1. *FEA-tilan toiminnot.*

Toiminto	**Kuvaus**
Uusi tutkimus	Luo uuden tutkimuksen valituista 3d-mallin osista. Valinta voidaan tehdä joko ennen tai jälkeen toimintonapin painamista.
Pistekuorma	Luo pistekuorman valituille solmuille.
Viivakuorma	Luo viivakuorman käyttäjän valitseman alku- ja loppusolmun välille.
Momentti	Luo pistemomentin valituille solmuille.
Pakkosiirtymä	Luo pakkosiirtymän valituille solmuille. Valittujen solmujen tulee olla tuettuja annetun pakkosiirtymän suunnassa.
Lämpökuorma	Luo lämpötilan muutoksesta aiheutuvan kuormituksen valituille tutkimuksen osille. Mahdollisuus antaa myös lämpötilagradientti poikkileikkauksen pääsuunnissa.
Ratkaise	Ratkaisee aktiivisena olevan tutkimuksen, jonka jälkeen käyttäjä voi tarkastella tuloksia tulosdialogissa.
Kuormakaaviot	Luo kuormakaavion valituista kuormista.
Nurjahduspituudet	Käynnistää nurjahduspituuksien määrittelydialogin, jossa käyttäjä voi asettaa nurjahduspituuksien arvoja halutuille solmuväleille.
Kapasiteetin tarkistus	Käynnistää kapasiteetin tarkistuksen. Kapasiteetin tarkistus suoritetaan ohjelmistoon kytketyn suunnittelustandardin mukaisesti käyttäjän antamien kuormakaavioiden, kuormitusyhdistelmien ja nurjahduspituuksien perusteella.
OK	Palataan 3d-mallinnustilaan säilyttäen FEA-tilassa tehdyt muutokset.
Poistu	Palataan 3d-mallinnustilaan säilyttämättä FEA-tilassa tehtyjä muutoksia.



Kuva 25. *Palkkien tilannekohtainen valintanauha.*

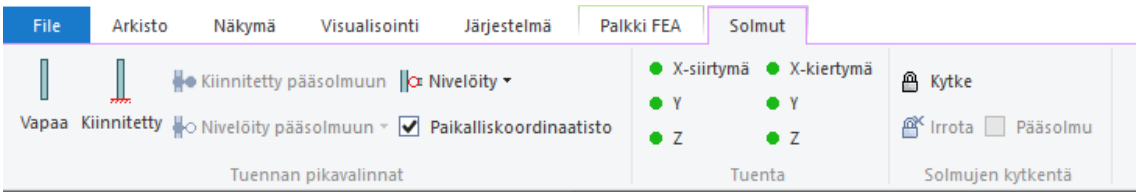


Kuva 26. *Palkkien tilannekohtainen valikko 3d-mallin ikkunassa.*

Kuvissa 25 ja 26 näkyvät tilannekohtaiset toiminnot yhden tai useamman 3d-mallin profiilin ollessa valittuina FEA-tilassa. Toimintojen kuvaukset ovat koottuna taulukkoon 2.

Taulukko 2. *Tilannekohtaiset toiminnot profiileja ollessa valittuina.*

Toiminto	**Kuvaus**
Ominaisuudet	Avaa ominaisuusdialogin, jossa valittujen profiilien poikkileikkaus- ja materiaalio minaisuuksia voidaan tarkastella ja muokata.
Lisää tutkimukseen	Lisää valitut profiilit aktiiviseen tutkimukseen.
Poista tutkimuksesta	Poistaa valitut profiilit aktiivisesta tutkimuksesta.
Lisää solmu	Käynnistää toiminnon, jossa käyttäjä voi osoittaa profiilin neutraaliakselilta paikan, johon lisätään uusi solmu.
Valitse solmut	Valitsee kaikki valittuina olevien profiilien solmut.
Kytke solmuja	Käynnistää toiminnon, jolla käyttäjä voi kytkeä valittujen profiilien lähimpiä solmuja toisiinsa annetulla toleranssilla ja kytkennän esiasetuksella.



Kuva 27. *Solmujen tilannekohtainen valintanauha.*

Kuvassa 27 näkyvät tilannekohtaiset toiminnot yhden tai useamman solmun ollessa valittuina FEA-tilassa. Solmujen tilannekohtaiset toiminnot liittyvät ainoastaan niiden tuentaan, kytkentään ja koordinaatistoon.

Solmujen tuenta voidaan asettaa vapausastekohtaisesti valintanauhan Tuenta-ryhmän painikkeilla. Solmuja voidaan kytkeä toisiinsa vapausastekohtaisesti orjuuttamalla [17]. Valittu solmu voidaan asettaa kytketyn solmuryhmän pääsolmuksi valintanauhan Solmujen kytkentä -ryhmän Pääsolmu-valintaruudulla. Ei-kytketyn solmun tai kytketyn pääsolmun vapausasteiden tuentavaihtoehtoja ovat vapaa ja tuettu. Kytketyn orjasolmun vapausasteiden tuentavaihtoehtoja ovat vapaa ja kytkentä pääsolmuun.

Valintanauhan Tuennan pikavalinnat -ryhmän painikkeilla voidaan asettaa solmuille nopeasti tyypillisimpiä tuentavaihtoehtoja, joiden kuvaukset on koottu taulukkoon 3. Ryhmän Paikalliskoordinaatisto-valintaruutu asettaa solmun koordinaatistoksi palkin poikkileikkauksen pääkoordinaatiston. Muutoin solmun koordinaatisto on samansuuntainen 3d-mallin koordinaatiston kanssa.

Taulukko 3. *Tuennan pikavalinnat.*

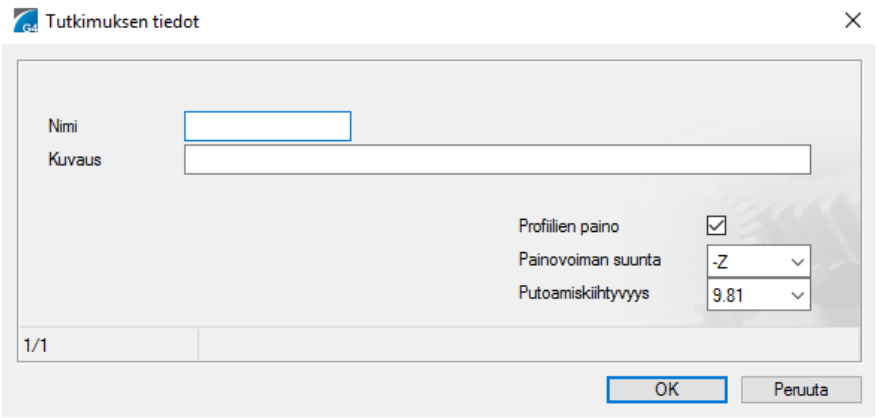
Pikavalinta	**Kuvaus**
Vapaa	Kaikki vapausasteet vapautettu. Mahdollinen kaikille solmuille.
Kiinnitetty	Kaikki vapausasteet tuettu. Mahdollinen kaikille ei-kytketyille solmuille ja kytketyille pääsolmuille.
Nivelöity	Siirtymävapausasteet tuettu, mutta osa kiertymävapausasteista vapautettu. Alasvetovalikon vaihtoehdoilla määritellään, mitkä kiertymävapausasteet ovat vapaina. Vaihtoehdot näkyvät kuvassa 28.
Kiinnitetty pääsolmuun	Kaikki vapausasteet kiinnitetty pääsolmuun. Mahdollinen kytketyille orjasolmuille.
Nivelöity pääsolmuun	Siirtymävapausasteet kiinnitetty pääsolmuun, mutta osa kiertymävapausasteista vapautettu. Alasvetovalikon vaihtoehdoilla määritellään, mitkä kiertymävapausasteet ovat vapaina. Vaihtoehdot näkyvät kuvassa 28.

Kuva 28. *Solmun nivelöinnin pikavalinnan vaihtoehdot.*

Kokonaisten tutkimusten hallintaan liittyvät toiminnot käynnistetään tutkimuspuun tilannekohtaisesta valikosta valitsemalla halutut tutkimukset ja painamalla hiiren oikeaa painiketta. Tutkimusten hallintaan liittyvien toimintojen kuvaukset ovat koottuna taulukkoon 4.

Taulukko 4. *Tilannekohtaiset toiminnot aktiivisen tutkimuksen ollessa valittuna.*

Toiminto	**Kuvaus**
Ratkaise	Ratkaisee aktiivisena olevan tutkimuksen, jonka jälkeen käyttäjä voi tarkastella tuloksia tulosdialogissa.
Valitse osat	Valitsee kaikki 3d-mallin osat, jotka kuuluvat tutkimukseen.
Muokkaa	Käynnistää tutkimuksen muokkausdialogin, jossa käyttäjä voi muokata tutkimuksen ominaisuuksia. Dialogi on esitetty kuvassa 29.
Kopioi	Kopioi kaiken tutkimuksen sisällön uuteen tutkimukseen. Ei koske 3d-mallin osia, vaan ainoastaan laskentamalliin liittyvää tietoa.



Kuva 29. *Tutkimuksen tiedot -dialogi.*

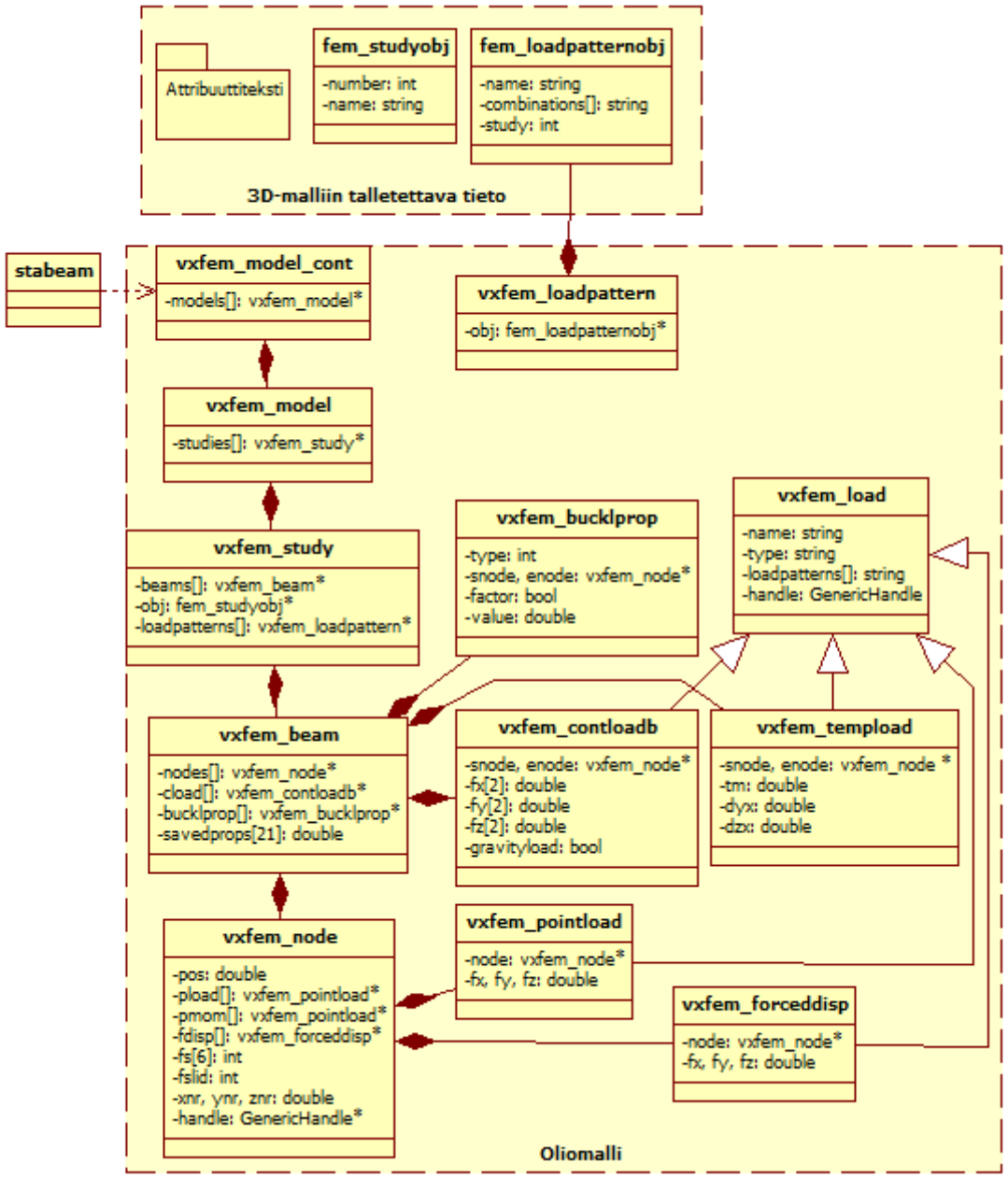
6.2 Tietorakenne

Tässä luvussa perehdytään tarkemmin kehärakenteiden lujuuslaskentamoduulin käyttöliittymän takana olevaan tietorakenteeseen sekä tiedon tallentamiseen ohjelmakoodin tasolla. Käyttöliittymän ohjelmakoodi on toteutettu MVC-arkkitehtuurin suunnitteluperiaatteita noudattaen.

6.2.1 Tiedon tallennus ja oliomalli

Lujuuslaskentaan liittyvät tiedot kuten elementtiverkko, kuormitukset ja reunaehdot muodostavat laskentamallin varsinaisen mallinnetun geometrian rinnalle. Laskentamallin tiedot tallennetaan Vertex-ohjelmistolla luotujen 3d-geometriaa sisältävien mallien sisään attribuuttitekstinä. Attribuuttiteksti tarkoittaa 3d-mallin yksittäiseen osaan liitettyä merkkijonoa, joka tallentuu kovalevylle mallitiedostoon osan mukana.

Laskentamallin rakentaminen suoraan mallin osien attribuuttitekstejä muuttamalla olisi hidasta, sillä tällöin jouduttaisiin tekemään paljon hankalia operaatioita pitkille merkkijonoille. Laskentamallin tieto on pääosin lukuarvoja, joten sitä on helpompi ja tehokkaampi käsitellä tietokoneen muistissa kokonais- ja liukulukujen avulla. Laskentamallin tiedon käsittelyä varten päätettiin toteuttaa oliomalli, joka luodaan 3d-mallin osien attribuuttiteksteihin tallennettujen tietojen perusteella. Oliomalli luodaan käyttäjän siirtyessä FEA-tilaan. Käyttäjän poistuessa FEA-tilasta oliomallin tieto siirretään takaisin 3d-mallin osille attribuuttitekstiksi riippuen siitä, haluaako käyttäjä tallentaa FEA-tilassa tehdyt muutokset vai ei.



Kuva 30. *Oliomalli ja 3d-malliin talletettava tieto.*

Kuvassa 30 on esitetty yksinkertaistettuna luokkakaaviona laskentamallin toteuttava tietorakenne. 3d-malliin talletettavasta tiedosta luodaan kuvan mukainen oliomalli käyttäjän siirtyessä FEA-tilaan. Lujuusanalyysin suorittavat laskentamoottorin algoritmit on koottu `stabeam`-luokkaan. Lähtötiedot laskentamoottori saa oliomallilta. Oliomallin keskeisimmät osat ovat seuraavat:

- **`vxfem_model_cont`-luokka,** joka toimii säiliönä `vxfem_model`-luokan olioille.
- **`vxfem_model`-luokka**, joka sisältää listan tutkimuksista, jotka on luotu yhdelle 3d-mallille.
- **`vxfem_study`-luokka,** joka sisältää listat yhteen tutkimukseen kuuluvista `vxfem_beam`-luokan palkkeja kuvastavista olioista sekä

vxfem_loadpattern-luokan kuormakaavioita kuvastavista olioista. Luokan jäsenmuuttujana on myös viittaus 3d-malliin talletettavaan fem_studyobj-luokan olioon.

- **vxfem_beam-luokka,** jonka oliot kuvastavat laskentamallin palkkeja. Luokka sisältää listat solmuista, viivakuormista sekä nurjahduspituuksista. Luokan jäsenmuuttujana on myös liukulukutaulukko, johon on koottu käyttäjän syöttämät palkin materiaali- ja poikkileikkausarvot.
- **vxfem_node-luokka,** jonka oliot kuvastavat laskentamallin solmuja. Luokka sisältää listat erityyppisistä solmukuormista sekä alla olevaan taulukkoon 5 kootut jäsenmuuttujat.

Taulukko 5. *vxfem_node-luokan tietosisältö.*

Jäsenmuuttuja	Tyyppi	Sisältö
pos	liukuluku	Ilmaisee solmun suhteellisen paikan palkilla välillä 0.0-1.0
fs[6]	kokonaislukutaulukko	Ilmaisee solmun kiinnitykset vapausasteittain. Kolme ensimmäistä taulukon alkiota viittaavat siirtymiin ja kolme jälkimmäistä kiertymiin. Alkioiden arvojen merkitys on seuraava: • 0 = vapaa • 1 = kiinnitetty • 2 = orjuutettu pääsolmuun
fslid	kokonaisluku	Ilmaisee orjuutettujen solmujen ryhmänumeron. Alkion arvon merkitys on seuraava: • 0 = solmua ei orjuutettu • >0 = solmu on pääsolmu • <0 = solmu on orjasolmu
xnr, ynr ja znr	liukuluku	Ilmaisevat solmun paikallisen koordinaatiston kierron suhteessa 3d-mallin koordinaatistoon asteina.
handle	osoitin	Viittaus 3d-grafiikkaobjektin kahvaan

Laskennassa tarvittavat tiedot solmujen ja palkkien sijainneista ja asennoista saadaan 3d-mallin geometriasta, joten niitä ei tarvitse tallentaa laskentamallin tietojen yhteyteen. Samoin myös materiaali- ja poikkileikkausominaisuuksien oletusarvot saadaan 3d-mallista.

Kaikki laskentamallin erityyppiset kuormitukset on kuvattu oliomallissa abstraktista kantaluokasta vxfem_load perittyinä aliluokkina. Kantaluokka vxfem_load sisältää kaikille kuormille yhteistä tietoa, kuten kuorman nimen, listan kuormakaavioista sekä viittauksen kahvaan. Kahvalla tarkoitetaan liityntää 3d-mallissa esitettävään grafiikkaan, jota esitellään enemmän luvussa 6.2.2. Erityyppisiä kuormituksia kuvaavia luokkia ovat seuraavat:

- **`vxfem_contloadb`-luokka,** joka kuvastaa kahden solmun välistä viivakuormaa. Luokan jäsenmuuttujina ovat osoittimet alku- ja loppusolmuihin, viivakuorman arvo alku- ja loppusolmulla komponenteittain sekä totuusarvo, joka kertoo, onko viivakuorma palkin omasta painosta johtuva.
- **`vxfem_tempload`-luokka,** joka kuvastaa kahden solmun välistä lämpötilakuormaa. Luokan jäsenmuuttujina ovat lämpötilanmuutos tällä solmuvälillä sekä lämpötilagradientin arvo poikkileikkauksen pääsuunnissa.
- **`vxfem_pointload`-luokka,** joka kuvastaa joko pistekuormaa tai pistemomenttia. Luokan jäsenmuuttujina ovat kuormaan liittyvä solmu sekä voiman tai momentin arvo komponenteittain.
- **`vxfem_forceddisp`-luokka,** joka kuvastaa solmun pakkosiirtymää. Luokan jäsenmuuttujina ovat siirtymään liittyvä solmu sekä siirtymän arvo komponenteittain.

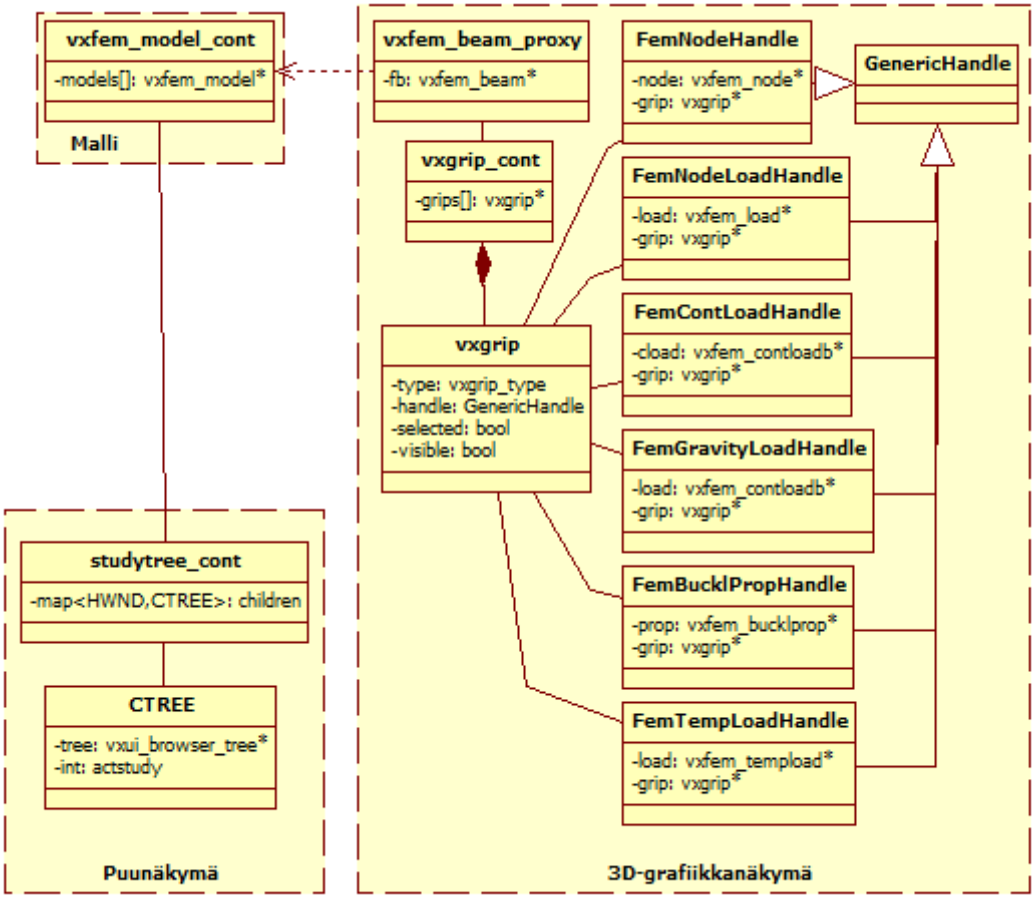
Kaikki kuormien arvot luokkien jäsenmuuttujissa on esitetty komponenteittain kuormaan liittyvän solmun paikalliskoordinaatistossa.

3d-malliin tallennetaan attribuuttitekstin lisäksi kuvan 30 mukaisesti seuraavien luokkien olioita:

- **`fem_studyobj`-luokka,** joka tallentaa yhden tutkimuksen ominaisuudet 3d-malliin. Tallennettavia ominaisuuksia ovat
  - tunnistenumero,
  - nimi,
  - kuvaus,
  - painovoiman vaikutus,
  - painovoiman suunta ja
  - putoamiskiihtyvyys.
- **`fem_loadpatternobj`-luokka**, joka tallentaa yhden kuormakaavion ominaisuudet 3d-malliin. Tallennettavia ominaisuuksia ovat nimi ja kyseiseen kuormakaavioon käytettävät kuormitusyhdistelmät.

6.2.2 Näkymät laskentamalliin

FEA-tilassa käyttäjällä on kaksi erilaista näkymää käsiteltävään laskentamalliin: 3d-mallissa näkyvät grafiikkaobjektit sekä tutkimuspuu. Kummankin näkymän kautta laskentamalliin kohdistetut toiminnot ovat toteutukseltaan täysin samoja.



Kuva 31. *Näkymät laskentamalliin.*

Kuvassa 31 on esitetty yksinkertaistettuna luokkakaaviona eri näkymät käsiteltävään laskentamalliin. Puunäkymä koostuu seuraavista osista:

- **`studytree_cont`-luokka,** joka toimii puunäkymän ikkunoiden säiliönä Vertex-sovelluksessa avoinna oleville 3d-malleille. Jokaista avoinna olevaa 3d-mallia kohden on oma puunäkymänsä, joka on näkyvillä riippuen siitä, onko käyttäjä siirtynyt FEA-tilaan vai ei. Tämän luokan jäsenmuuttujana on `std::map`-luokan säiliö, joka kytkee toisiinsa `HWND`-tyyppisen [10] 3d-mallin ikkunan kahvan sekä `CTREE`-luokan olion.
- **`CTREE`-luokka,** joka toimii yhden puunäkymän toteuttavana luokkana. Se sisältää viitteen `vxui_browser_tree`-luokan olioon, joka sisältää varsinaisen toiminnallisuuden toteutuksen puunäkymälle. `CTREE`-luokan jäsenmuuttujana on myös kokonaisluku, joka ilmaisee aktiivisena olevan tutkimuksen tunnistenumeron kyseisessä puunäkymässä.

Puunäkymän varsinaisen toiminnallisuuden sisältävää `vxui_browser_tree`-luokkaa on käytetty Vertexin ohjelmistoissa monissa muissakin paikoissa. Tässä tapauksessa sen toimintaan tarvitsi tehdä vain pieniä muutoksia, joilla se saatiin valjastettua toimimaan

tutkimuspuun tarpeiden mukaan. Puunäkymän alkioiden läpikäymiseen käytetään `vxui_browser_tree`-luokan toteutuksessa Visitor pattern -suunnittelumallia. Puunäkymä saa sisältönsä `vxfem_model`-luokan oliolta, ja tiedonsiirto tapahtuu XML-tiedoston välityksellä. Pohjimmiltaan `vxui_browser_tree`-luokka käyttää puunäkymän luomiseen Windows API -ohjelmointirajapinnan tarjoamia palveluja [11].

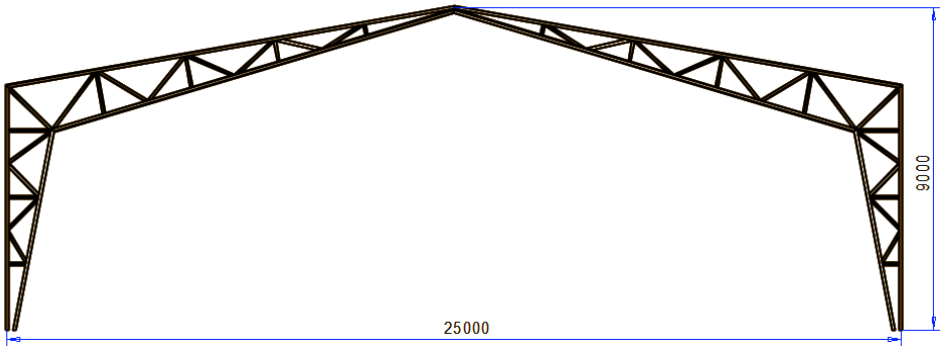
Kuvassa 31 esitellyn 3d-grafiikkanäkymän toteutuksen osat ovat seuraavat:

- **`vxgrip_cont`-luokka,** joka toimii säiliönä 3d-mallissa oleville `vxgrip`-luokan olioille.
- **`vxgrip`-luokka,** joka kuvastaa yhtä 3d-mallissa olevaa grafiikkaobjektia. Luokan jäsenmuuttujina ovat totuusarvot, jotka kertovat onko olio valittu ja onko se näkyvissä. Lisäksi jäsenmuuttujana on tieto olion tyypistä, jolloin samasta luokasta luotuja olioita voidaan tyypittää niiden käyttötavan mukaan. Esimerkiksi palkkielementin solmua esittävä `vxgrip`-luokan olio on tyypiltään eri kuin viivakuormaa esittävä `vxgrip`-luokan olio. Luokalla on jäsenmuuttujana myös viittaus `GenericHandle`-luokan olioon.
- **`GenericHandle`-luokasta perityt luokat,** jotka huolehtivat 3d-grafiikkaobjektin ruudulle piirron päivittämisestä. `GenericHandle`-luokka on Vertexin sovelluksissa aiemminkin monissa paikoissa käytetty. Se toteuttaa mahdollisuuden tarttua 3d-mallissa olevaan olioon viemällä kursori sen päälle ja painamalla hiiren painiketta.
- **`vxfem_beam_proxy`-luokka,** joka sisältää `vxfem_beam`-luokan olioihin liittyviä metodeja, jotka eivät kuitenkaan muokkaa `vxfem_beam`-luokan olioiden sisältöä. Tällaisia metodeja ovat esimerkiksi laskentamallin palkkia kuvastavaan `vxfem_beam`-luokan olioon liittyvien `vxgrip`-luokan grafiikkaobjektien luonti ja tuhoaminen. Tämän luokan metodien kautta tapahtuu myös varsinainen grafiikan piirto ruudulle, joka toteutetaan OpenGL-ohjelmointirajapinnan tarjoamien palveluiden avulla [7].

6.3 Käyttöesimerkki: kolminivelkehän lujuustarkastelu

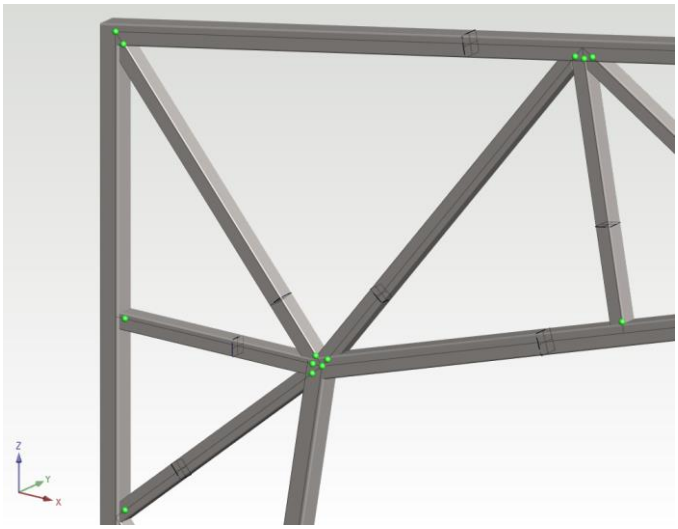
Tässä esimerkissä tarkastellaan kuvassa 32 esitettyä tyypillistä Vertex G4 -ohjelmistolla mallinnettua kolminivelkehää. Tavoitteena on tarkastella rakenteen maksimisiirtymiä sekä Von Mises -vertailujännityksiä, kun kehän yläpaarteilla vaikuttavat 6 kN/m suuruiset viivakuormat. Lisäksi rakenteen oma paino huomioidaan laskennassa. Kehän liitokset ovat hitsattuja pois lukien paarteiden liitoskohta, jossa on nivel.

Kehän profiilit ovat poikkileikkaukseltaan neliön muotoisia RHS-putkipalkkeja ja niiden materiaalina on S355 rakenneteräs. Paarteiden poikkileikkauksen korkeus on 100 mm ja seinämänpaksuus 5 mm. Diagonaalien poikkileikkauksen korkeus on 80 mm ja seinämänpaksuus 4 mm.



Kuva 32. *Tarkasteltava kolminivelkehä.*

6.3.1 Tutkimuksen luonti ja osien kytkentä



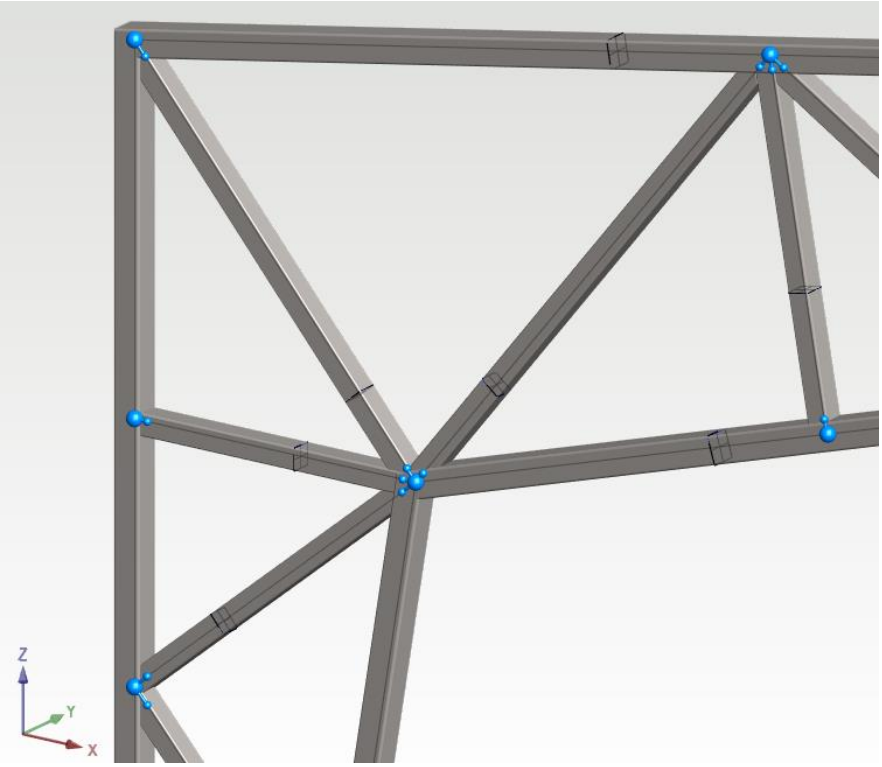
Kuva 33. *Rakenne ilman solmujen kytkentää.*

Analyysi aloitetaan siirtymällä 3d-mallinnustilasta FEA-tilaan ja tekemällä kehän osista uusi tutkimus. Oletuksena uuden tutkimuksen osille luodaan yksi palkkielementti, eli profiilien päihin syntyy solmut kuvan 33 osoittamalla tavalla.

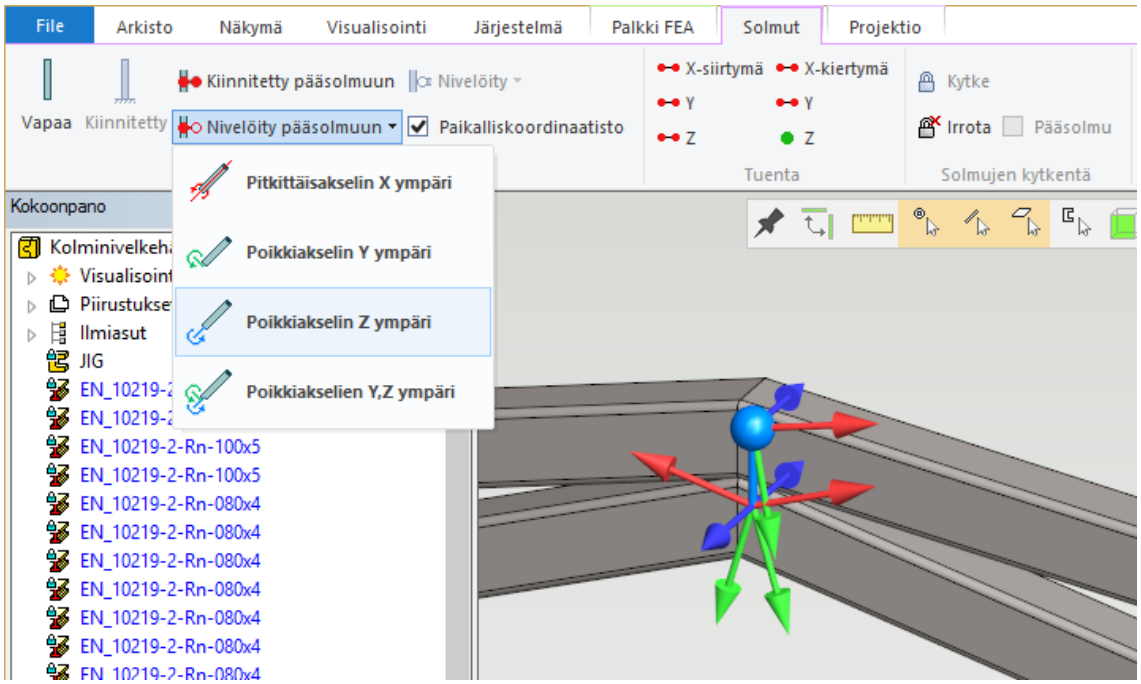
Koska kehän profiilit on liitetty toisiinsa hitsaamalla, tulee laskentamallin palkkielementtien välille saada luotua jäykkiä kytkentöjä. Tässä esimerkissä siis paarteille tulisi lisätä solmuja sopivasti siten, että ne muodostuvat useammasta palkkielementistä. Paarteiden palkkielementtien päätysolmut tulisi olla diagonaalien ja paarteiden neutraaliakseleiden leikkauskohdissa.

Tarvittavien solmujen luominen ja niiden kytkeminen voidaan tehdä käsin, mutta se on varsin työlästä. Tämä työvaihe voidaan automatisoida palkkien tilannekohtaisen valintanauhan Kytke solmuja -toiminnolla. Kyseinen toiminto muodostaa automaattisesti valittujen profiilien toisiaan lähellä olevista solmuista kytkettyjä solmuryhmiä. Toiminto myös luo automaattisesti solmun profiilille, mikäli profiilin neutraaliakselin läheisyydessä on solmu. Etäisyystoleranssi, jolla kytketty solmuryhmä tai profiilin uusi solmu luodaan, on käyttäjän annettavissa samoin kuin muodostettavien solmuryhmien orjasolmujen kytkentätyyppi pääsolmuun.

Kuvasta 34 nähdään kuinka Kytke solmuja -toiminnon muodostamat kytketyt solmuryhmät ovat muodostuneet. Nyt laskentamallin palkkielementit ovat liittyneet toisiinsa jäykillä liitoksilla.



Kuva 34. *Rakenne solmujen kytkennän jälkeen.*

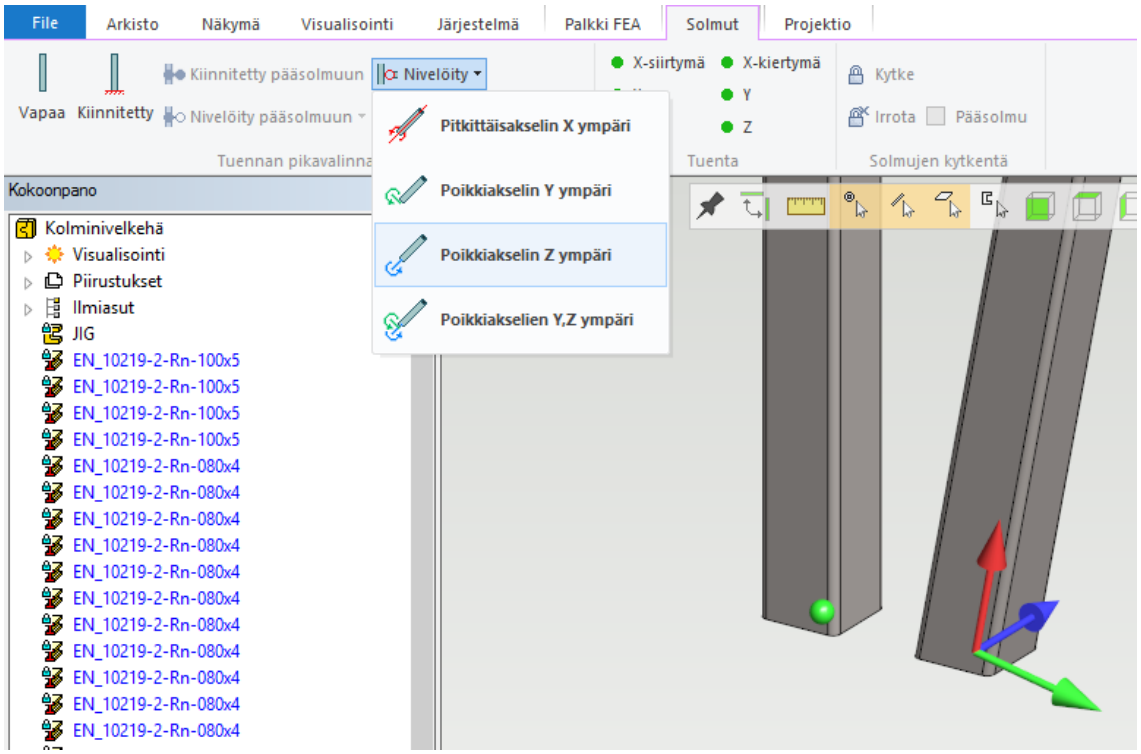


Kuva 35. *Kehän yläpaarteiden nivelöinti.*

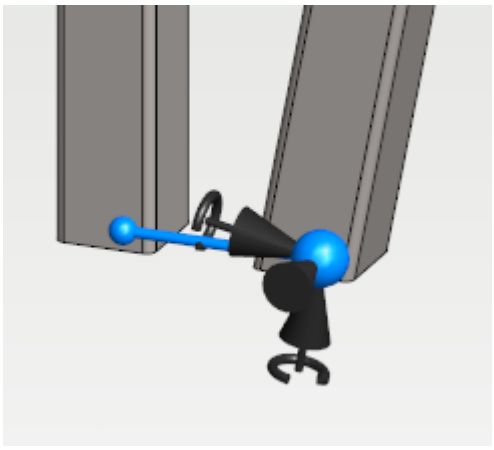
Yläpaarteiden liitoskohta on automaattisen solmujen kytkennän jäljiltä jäykkä liitos. Liitoksen muuttaminen nivelelliseksi onnistuu valitsemalla kytketyn solmuryhmän orja-solmut ja asettamalla niille solmujen tilannekohtaisesta valintanauhasta Tuennan pika-valinnat -ryhmästä toiminto Nivelöity pääsolmuun poikkiakselin Z-ympäri kuvan 35 osoittamalla tavalla. Kuvassa näkyvät koordinaatistot kertovat valittujen solmujen pai-kalliskoordinaatiston asennon. Kaikkien valittujen solmujen paikalliskoordinaatiston Z-akseli (sininen nuoli) on yhdensuuntainen kehän tason normaalin kanssa. Täten yläpaar-teiden liitoskohtaan syntyy kehän tason normaalin ympäri pyörivä nivel.

6.3.2 Tuenta

Kehän jalkojen alapäät ovat nivelöityjä. Oletetaan, että kehän jalat kiertyvät liitoksessa sisemmän profiilin päätysolmun ympäri, eli kuvassa 36 valitun solmun paikalliskoordinaatiston Z-akselin (sininen nuoli) ympäri. Asetetaan valittu solmu nivelöidyksi poikkiakselin Z ympäri. Kytketään vielä jalan alapään molempien profiilien päätysolmut keskenään jäykästi kaikkien vapausasteiden osalta. Tämä vastaa käytännössä siis tilannetta, että profiilien päät olisivat yhdistetty toisiinsa esimerkiksi hitsatulla lattaraudalla. Kuvasta 37 nähdään jalkojen alapäiden solmujen kytkentä ja tuenta. Mustat symbolit merkitsevät vapausastekohtaisia tuentoja ja siniset solmut kytkentää solmujen välillä.

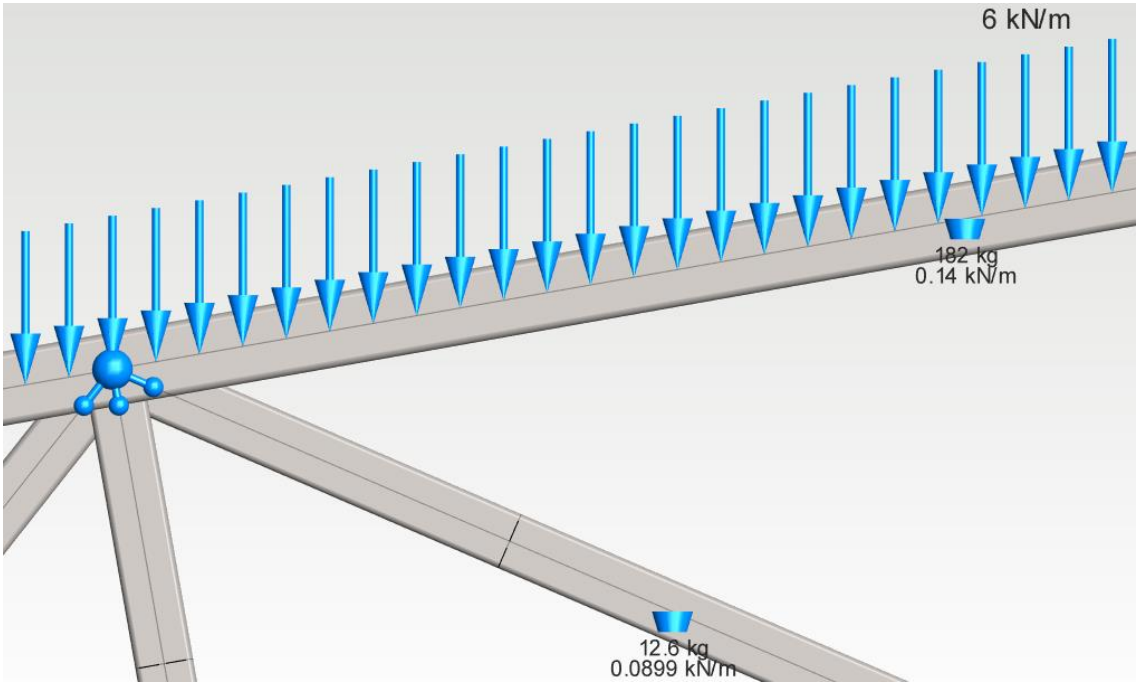


Kuva 36. *Kehän jalkojen alapäiden tuenta.*



Kuva 37. *Kehän jalkojen alapäiden tuenta ja kytkentä.*

6.3.3 Kuormitus

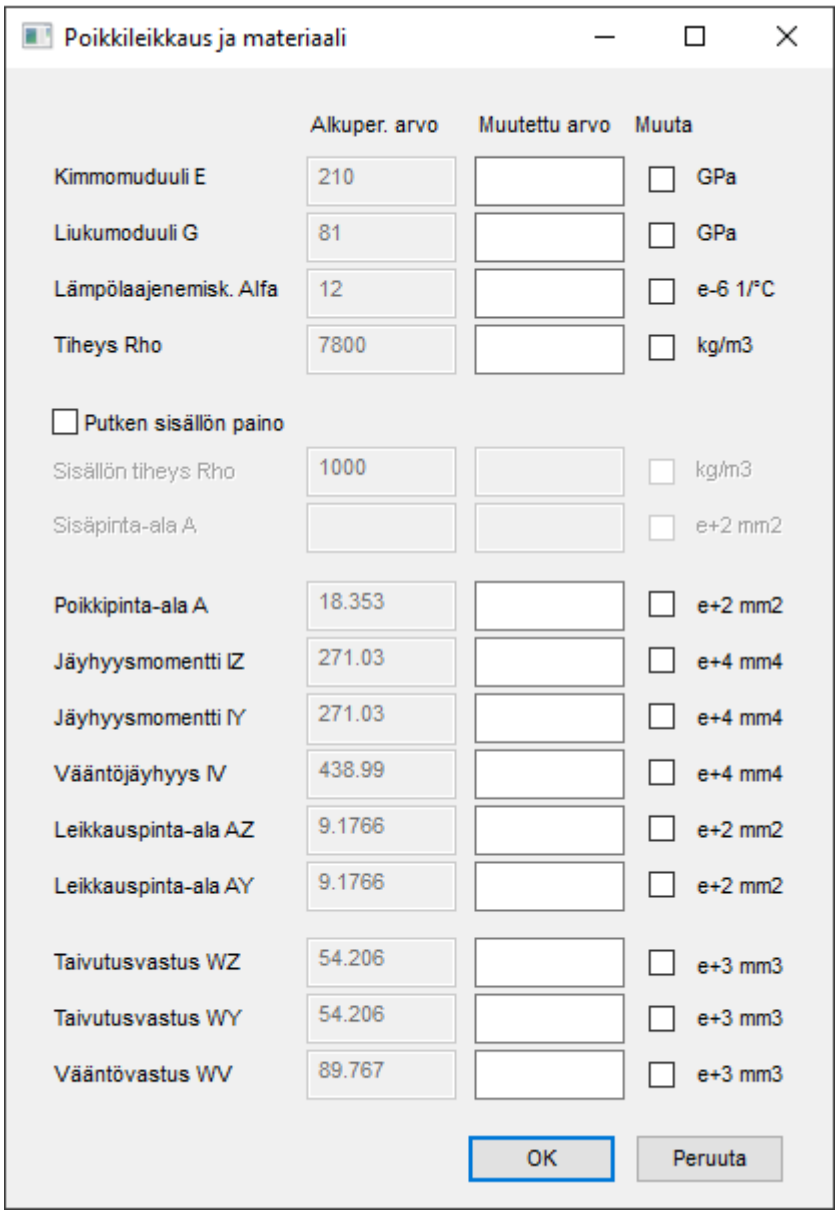


Kuva 38. *Kehän yläpaarteiden kuormitus.*

Kehää kuormittaa osien oman painon lisäksi myös 6 kN/m suuruinen viivakuorma yläpaarteilla. Viivakuormat asetetaan käynnistämällä FEA-tilan valintanauhasta toiminto Viivakuorma. Toiminnon käynnistyttyä käyttäjän tulee valita alku- ja loppusolmut lisättävälle viivakuormalle. Solmujen osoittamisen jälkeen käyttäjä syöttää viivakuormalle arvon alku- ja loppusolmuilla sekä suunnan. Kuormalle on mahdollista antaa myös nimi, joka näkyy tutkimuspuussa. Kuvassa 38 näkyy kehän yläpaarteelle asetettu viivakuorma. Puolikartion muotoiset siniset symbolit kuvastavat profiilin omaa painoa.

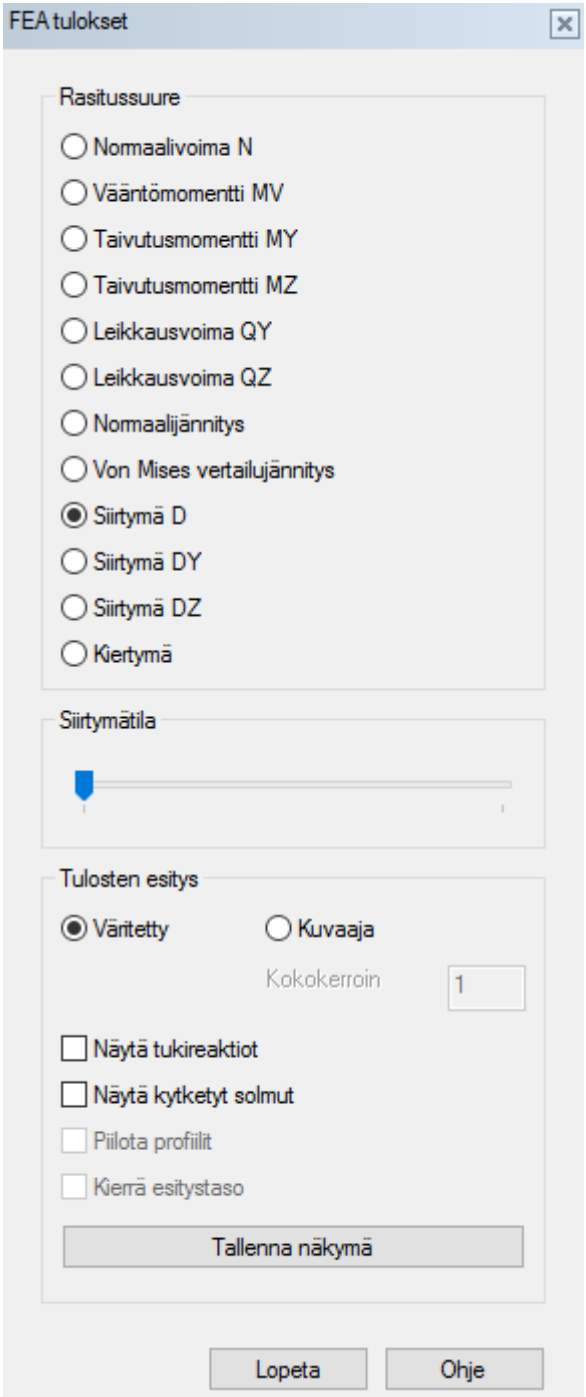
6.3.4 Tulokset

Kolminivelkehän laskentamalli on nyt reunaehtojen ja kuormitusten osalta valmis ratkaistavaksi. Vielä ennen ratkaisua on hyvä tarkistaa osien materiaali- ja poikkileikkausominaisuudet. Ne saadaan näkyville tuplaklikkaamalla haluttua osaa, mikä käynnistää dialogin, jossa käyttäjä voi tarkastella osakohtaisia ominaisuuksia. Esimerkiksi kehän yläpaarteella olevan RHS-putkipalkin 3d-mallin geometriasta lasketut poikkileikkausarvot ja 3d-mallin nimiketiedoista saadut materiaaliominaisuudet näkyvät kuvassa 39 olevassa dialogissa. Kaikki materiaali- ja poikkileikkausarvot ovat käyttäjän ylikirjoitettavissa, mikäli halutaan esimerkiksi käyttää standardissa taulukoituja arvoja.



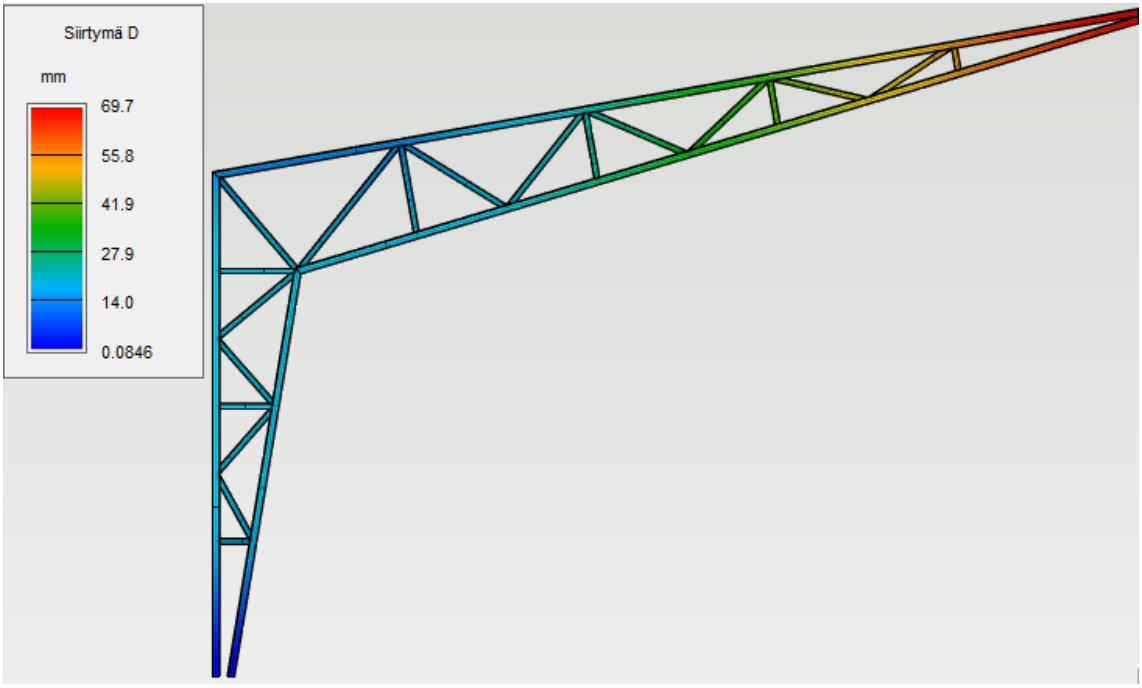
Kuva 39. *Tutkimuksen osan poikkileikkaus- ja materiaaliominaisuudet.*

Aktiivinen tutkimus ratkaistaan FEA-tilan valintanauhan Ratkaise -toiminnolla. Ratkaisun valmistuttua käynnistyy kuvan 40 mukainen tulosdialogi, jonka toiminnoilla käyttäjä voi ohjata tulosten esitystapaa.

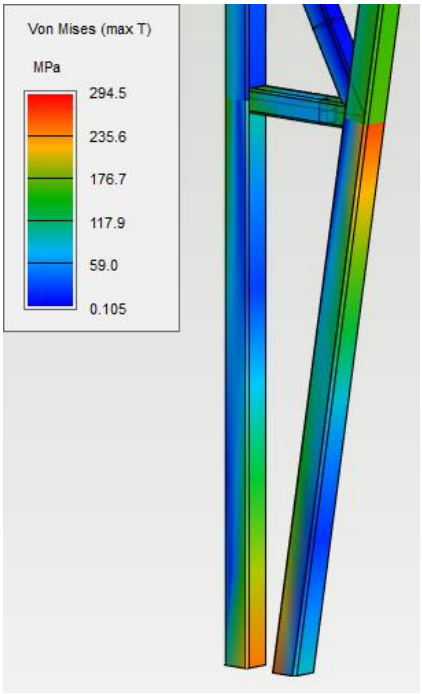


Kuva 40. *Tulosdialogi.*

Kuvassa 41 on esitetty puolikkaan kolminivelkehän siirtymätila ja kuvassa 42 Von Mises -vertailujännitys kehän jalan alapäässä, minne suurimmat jännitykset muodostuivat. Käyttäjä pystyy tarkastelemaan 3d-mallia tulosdialogin ollessa auki. Esimerkiksi mallin pyörittäminen ruudulla ja suurentaminen haluttuun kohtaan onnistuvat samaan tapaan kuin 3d-mallinnustilassakin.

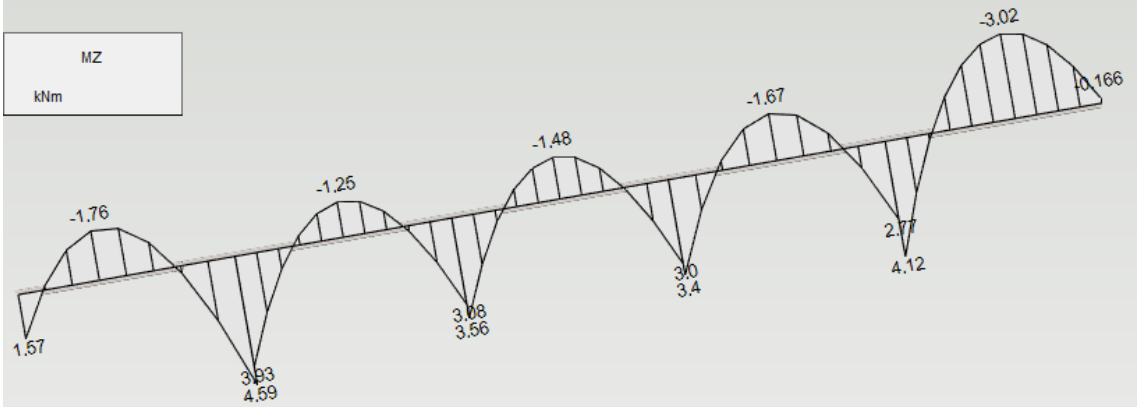


Kuva 41. *Kehän siirtymätila.*



Kuva 42. *Kehän jalan alapään Von Mises -vertailujännitys.*

Väritetyn tulosten esitystavan lisäksi rasitussuureet voidaan piirtää 3d-mallin päälle kaksiulotteisina kuvaajina. Tulokset voidaan myös rajata näytettäväksi ainoastaan halutuille tutkimuksen osille piilottamalla ennen ratkaisua ne osat, joita ei haluta mukaan tulosten esitykseen. Esimerkiksi kuvassa 43 on esitetty kehän yläpaarteen taivutusmomenttikuvio.



Kuva 43. *Kehän yläpaarteen taivutusmomenttikuvio.*

Suunnittelija pystyy tulosten perusteella nopeasti vertailemaan erilaisten rakennevaihtoehtojen käyttäytymistä kuormituksen alaisena. 3d-mallin geometria on muokattavissa normaaleilla Vertex-ohjelmistojen mallinnustyökaluilla, kun FEA-tilasta poistutaan. Tutkimusten laskentamallit pysyvät laskentakelpoisina vaikka 3d-mallin geometria muuttuu, kunhan mallin topologia säilyy ennallaan. Esimerkiksi profiilirakenteen poikkileikkausten vaihtamien toisiin säilyttää laskentamallin laskentakelpoisena. Mikäli 3d-mallin geometriaan tehdään suurempia muutoksia, esimerkiksi lisätään uusia osia, tulee laskentamalli luonnollisesti korjata laskentakelpoiseksi ainakin muuttuneilta osin.

7. YHTEENVETO

Tämän diplomityön tavoitteena oli Vertex CAD -ohjelmistoihin liitettävän kehärakenteiden lujuuslaskentamoduulin käyttöliittymän suunnittelu ja toteutus. Lujuuslaskentamoduuli on lisäoptio kolmelle eri Vertex CAD -suunnitteluohjelmistolle, jotka ovat

- Vertex G4 -mekaniikkasuunnitteluohjelmisto,
- Vertex G4 Plant -laitos- ja putkistosuunnitteluohjelmisto sekä
- Vertex BD -rakennussuunnitteluohjelmisto.

Kaikille näille kolmelle ohjelmistolle on tyypillistä, että niillä suunnitellaan erilaisia profiilirakenteita.

Vertex-ohjelmistojen lujuuslaskentaominaisuuksien uudistamiseen ryhdyttiin asiakastarpeesta. Usealta asiakkaalta oli tullut toiveita Vertex-suunnitteluohjelmistojen lujuuslaskentaominaisuuksien kehittämisestä. Myös kilpailevilla ohjelmistoilla on vastaavia CAD-suunnitteluohjelmaan liitettyjä lujuuslaskentamoduuleja, kuten esimerkiksi Dassault Systemesin SolidWorks Simulation [3].

Kehärakenteiden lujuuslaskentamoduulin käyttöliittymä toteutettiin Microsoft Windows -ympäristössä C++-ohjelmointikielellä osana suurempaa Vertex-ohjelmistokokonaisuutta. Käyttöliittymä toteutettiin MVC-ohjelmistoarkkitehtuurityyliä noudattaen. Mahdollisimman hyvän käyttäjäkokemuksen saavuttamiseksi tehtiin kirjallisuustutkimusta, jonka opastamana käyttöliittymän suunnittelussa ja toteutuksessa huomioitiin käyttäjäkeskeisen suunnittelun keskeisimpiä periaatteita.

Kehärakenteiden lujuuslaskentamoduuli rakentuu Vertexillä kehitetyn elementtimenetelmää hyödyntävän STAFRA-laskentamoottorin päälle. Lujuuslaskentamoduulin käyttöliittymä mahdollistaa rakenteen laskentamallin muodostamisen profiilirakenteen 3d-mallin geometrian perusteella. Solmuverkko voidaan luoda geometrian perusteella automaattisesti ja käyttäjä voi myös muokata sitä interaktiivisesti solmujen ja elementtien tasolla. Laskentamalli voidaan ratkaista käyttäjän antamilla tuentareunaehdoilla ja kuormituksilla. Laskentamalli koostuu Eulerin-Bernoullin palkkimallin mukaisista elementeistä. Ratkaisun tuloksista voidaan lukea laskentamallin rakenneosien rasitussuureita sekä siirtymiä. Tulokset voidaan esittää 3d-mallin geometriaan väritettyinä pintoina tai kaksiulotteisina kuvaajina. Tulosvaihtoehtoina ovat palkin

- normaalivoima,
- vääntömomentti,

- taivutusmomentti poikkileikkauksen 1. pääsuunnassa,
- taivutusmomentti poikkileikkauksen 2. pääsuunnassa,
- leikkausvoima poikkileikkauksen 1. pääsuunnassa,
- leikkausvoima poikkileikkauksen 2. pääsuunnassa,
- normaalijännitys poikkileikkauksen kaikissa pisteissä,
- Von Mises -vertailujännitys poikkileikkauksen kaikissa pisteissä,
- neutraaliakselin siirtymäresultantti,
- neutraaliakselin siirtymä poikkileikkauksen 1. pääsuunnassa,
- neutraaliakselin siirtymä poikkileikkauksen 2. pääsuunnassa sekä
- kiertymä.

Kehärakenteiden lujuuslaskentamoduulin kehityksen yhteydessä toteutettiin myös Kehärakenteiden mitoitus -lisäoptio Vertex BD -ohjelmistoon. Kehärakenteiden mitoitus -lisäoptiota myydään lisäominaisuutena tässä työssä käsitellylle kehärakenteiden lujuuslaskentamoduulille. Se tekee analysoitavalle rakenteelle rasitusten laskennan lisäksi myös kantavien rakenteiden suunnittelustandardien mukaisen mitoituksen. Vuonna 2016 julkaistavaan pääversioon tuettuna tulee olemaan Australiassa käytössä oleva suunnittelustandardi AS/NZS 4600:2005 Cold-formed steel structures.

Jatkokehitysideoita kehärakenteiden lujuuslaskentamoduuliin eteenpäin viemiseksi on tullut kehityksen yhteydessä paljon ja osaa ideoista on jo alettu toteuttaakin. Vertex G4 ja G4 Plant -asiakkailta on tullut toiveita ominaisuuksista, joilla pystyisi tekemään teräsrakenteille Eurocode 3 -suunnittelustandardin mukaisen mitoituksen. Yksi suurimmista kehityskohteista tulevaisuudessa tulee olemaankin Vertex BD -ohjelmistossa käytettävän Kehärakenteiden mitoitus -lisäoption ohjelmistoalustan yhteensovittaminen Vertex G4 ja G4 Plant -ohjelmistoihin. Tämän yhteensovittamisen jälkeen voidaan alkaa toteuttamaan Eurocode 3 -suunnittelustandardin mukaisen mitoituksen tuottavia ominaisuuksia Kehärakenteiden mitoitus -lisäoptioon, joka on tavoitteena tuoda tulevaisuudessa myös Vertex G4 ja G4 Plant -asiakkaiden käyttöön.

Toinen suuri jatkokehityskohde on Vertex BD -ohjelmiston Truss Engineering -lisäoption automaattisessa ristikkorakenteen mitoituksessa muodostettavan laskentamallin yhteensovittaminen tässä työssä käsitellyn Kehärakenteiden FEM-analyysin kanssa. Tavoitteena on saada Truss Engineering -lisäoptiolla muodostetun ristikkorakenteen laskentamalli käyttäjän muokattavaksi samoilla työkaluilla kuin Kehärakenteiden FEM-analyysissä.

LÄHTEET

[1] K.J. Bathe, *Finite Element Procedures in Engineering Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982, 735 p.

[2] J. S. Chabura, J. M. Leake, W. B. Hall, *Development of Instructional Software for Demonstrating CAD/FEA Integration Best Practices,* Department of General Engineering, University of Illinois at Urbana-Champaign, 2004, 9 p.

[3] SOLIDWORKS Simulation Packages, Dassault Systemes, 2016. Saatavissa (viitattu 13.9.2016): http://www.solidworks.com/sw/products/simulation/packages.htm

[4] E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns – Elements of Reusable Object-Oriented Software,* Addison-Wesley, 1995, 386 p.

[5] I. Haikala, J. Märijärvi, *Ohjelmistotuotanto*, Talentum Media Oyj, 2006, 440 s.

[6] STAFRA-3D -manuaali, Insinööritoimisto Lujuustekniikka Oy, 1980, 57 s.

[7] OpenGL API Documentation, The Khronos Group, 2016. Saatavissa (viitattu 23.6.2016): https://www.opengl.org/documentation/

[8] K. Koskimies, *Oliokirja*, Talentum Oyj, 2000, 422 s.

[9] R. Kouhia, M. Tuomala, *Johdatus mekaniikan ja sähkömagnetiikan numeerisiin menetelmiin*, luentomoniste, Tampereen teknillinen yliopisto, 2014, 417 s.

[10] hWnd Property, Microsoft, 2016. Saatavissa (viitattu 23.6.2016): https://msdn.microsoft.com/en-us/library/aa979055(v=vs.71).aspx

[11] Tree View (Windows), Microsoft, 2016. Saatavissa (viitattu 23.6.2016): https://msdn.microsoft.com/en-us/library/windows/desktop/bb759988(v=vs.85).aspx

[12] J. Nielsen, *Usability engineering,* Academic Press, 1993, 362 p.

[13] D. Norman, *The Design of Everyday Things – Revised and Expanded Edition*, Basic Books, 2013, 347 p.

[14] E. Oñate, *Structural Analysis with the Finite Element Method. Linear Statics. Volume 2. Beams, Plates and Shells*, Springer, 2013, 864 p.

[15] M. Rintala, J. Jokinen, *Olioiden ohjelmointi C++:lla,* Talentum, 2005, 466 s.

[16] B. Roith, A. Troll, F. Rieg, *Integrated Finite Element Analysis (FEA) in Three-dimensional Computer Aided Design Programs (CAD) – Overview and Comparison*, Department of Engineering Design and CAD, University of Bayreuth (Germany), 2007, 12 p.

[17] T. Salmi, K. Kuula, *Rakenteiden Mekaniikka*, Pressus Oy, Tampere, 2012, 464 s.

[18] B. Shneiderman, *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley, 1998, 639 p.

[19] 3D Mesh toolkit - Surface and volume meshing, Spatial Corp., 2016. Saatavissa (viitattu 1.6.2016): http://www.spatial.com/products/3d-mesh

[20] Vertex - Suunnitteluohjelmistot Tiedonhallintaohjelmistot 3D CAD PDM PLM, Vertex Systems Oy, 2016. Saatavissa (viitattu 24.5.2016): http://www.vertex.fi/web/fi/

[21] K. Väänänen, Käyttäjäkokemuksen perusteet -kurssin luentomoniste, Tampereen teknillinen yliopisto, 2015, 68 s.